Implications of M&S Foundations for V&V of Large-scale Complex Simulation Models

B.P. Zeigler, University of Arizona

and

H.S. Sarjoughian, Arizona State Univ.

Arizona Center for Integrative Modeling & Simulation

www.acims.arizona.edu

Foundation `02 Workshop

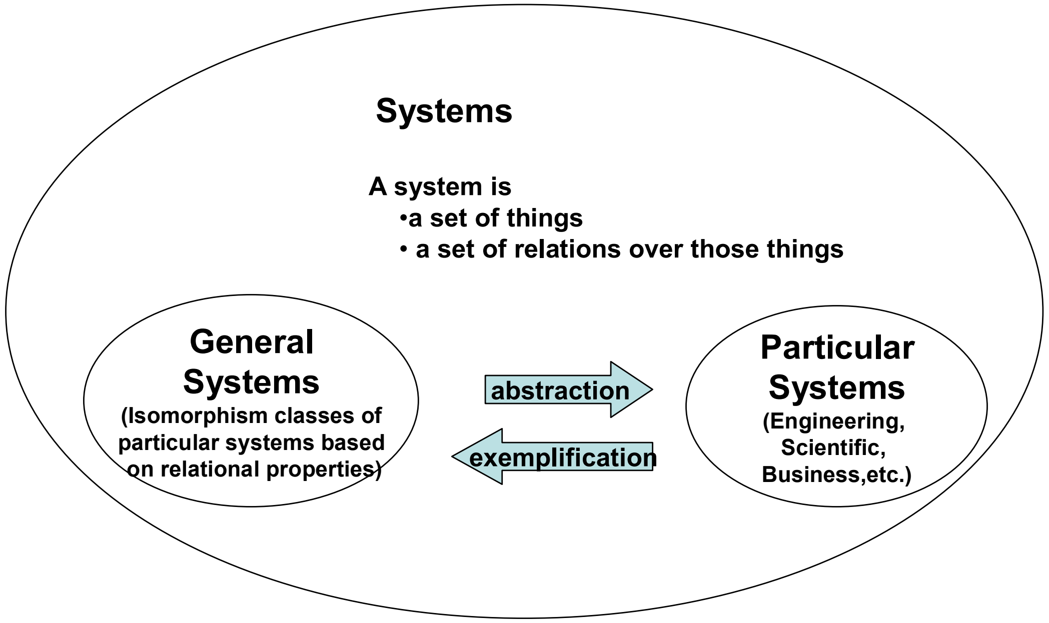
John Hopkins Univ./APL

Laurel, Maryland, USA

Outline

- Few general tools ease the demanding V&V process for large scale complex models
- There is a theory and framework to support the development of such tools
- Will review of the theory and framework
- Show how the theory-based concepts offer a systematic guideline to V&V in the various key steps of the HLA FEDEP
- Discuss the possibility of universal V&V tools for very large scale simulation models
  - based on computationally feasible domain-independent spaces
  - conceivable within the spirit of general systems theory
  - offer advantageous comparison of complex dynamical behaviors

General System Philosophy

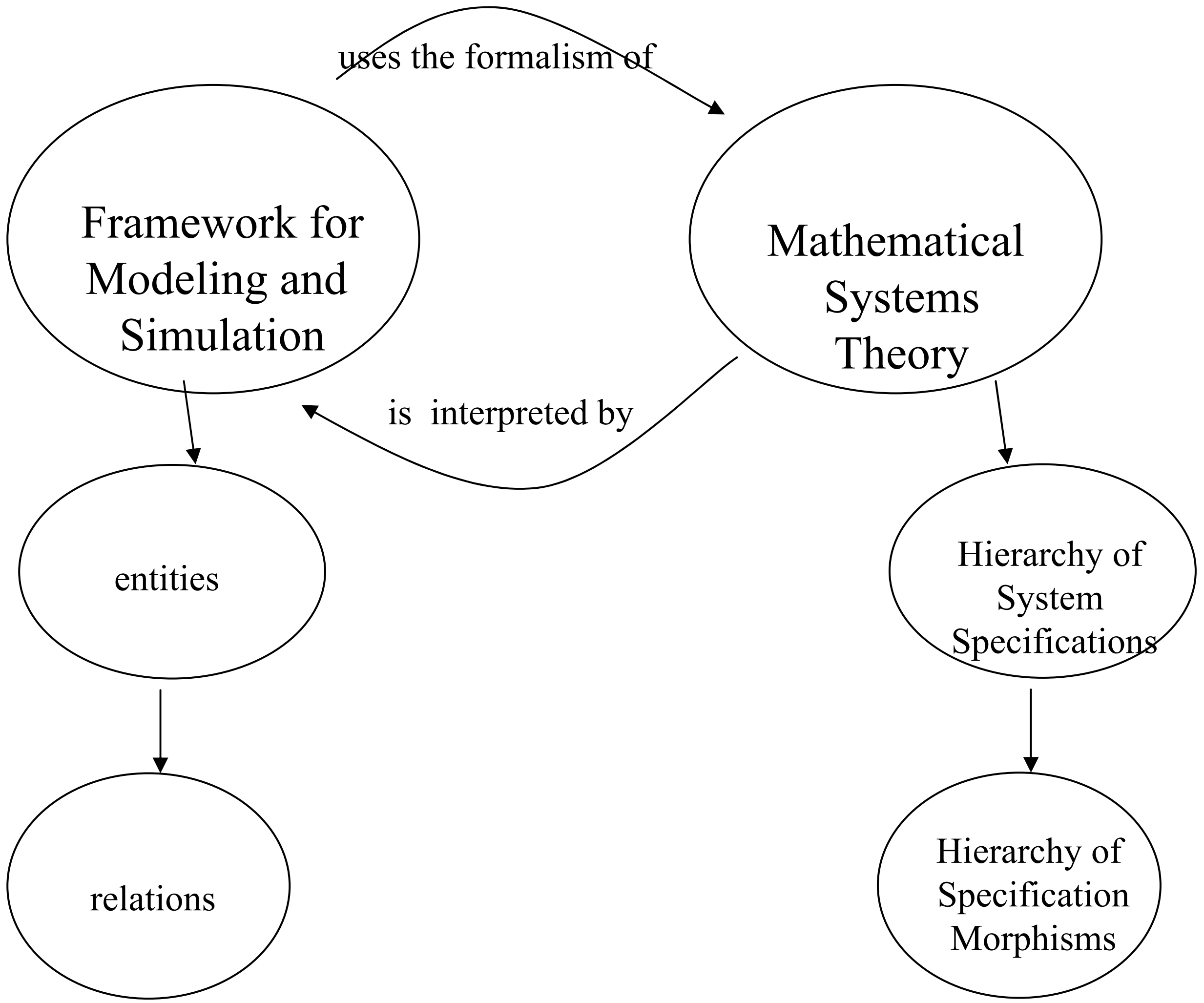

Systems
A system is
•a set of things
• a set of relations over those things
General Systems
(Isomorphism classes of particular systems based on relational properties)
abstraction
exemplification
Particular Systems
(Engineering, Scientific, Business,etc.)

- relational properties (focus on relations)
- domain independent
- interpretation free
- e.g. control theory, information theory
- theoretically based distinctions

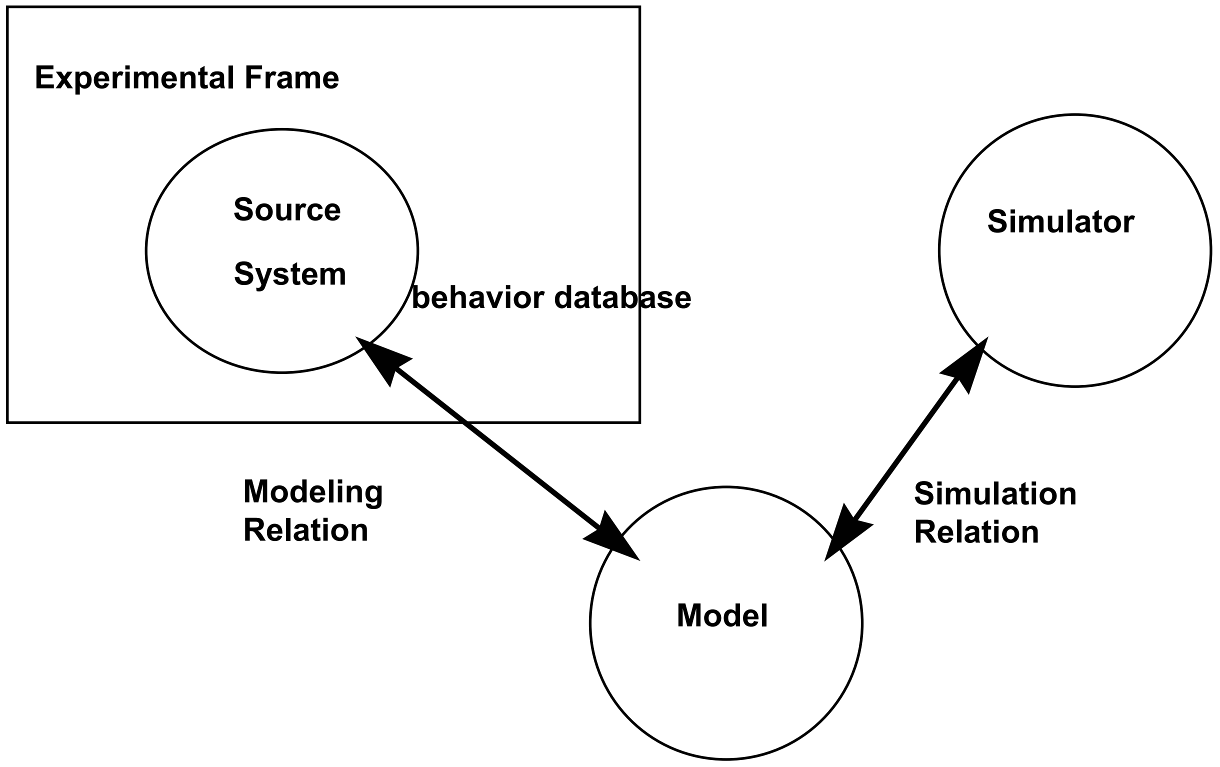
- constituent properties (focus on things)
- domain dependent
- interpretation dependent
- e.g. aeronautical control systems
  , business information systems
- experimentally based distinctions

George Klir, "Architecture of General Systems"

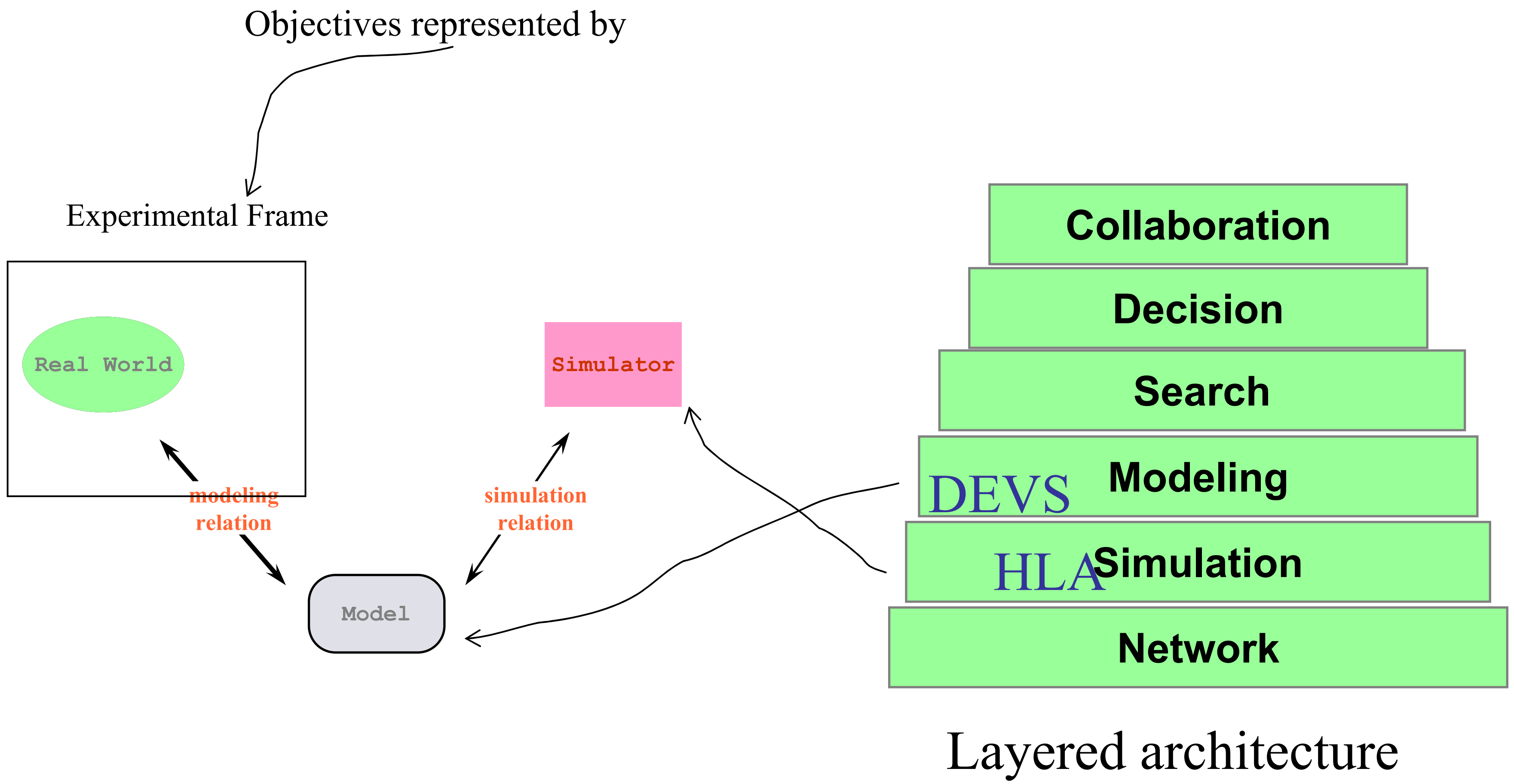
Modeling & Simulation/Systems Theory


uses the formalism of
Framework for Modeling and Simulation
Mathematical Systems Theory
is interpreted by
entities
relations
Hierarchy of System Specifications
Hierarchy of Specification Morphisms

Basic Entities and Relations in Modeling and Simulation



M&S Framework



Entities formalized as systems; relations as system morphisms

DEVS Background

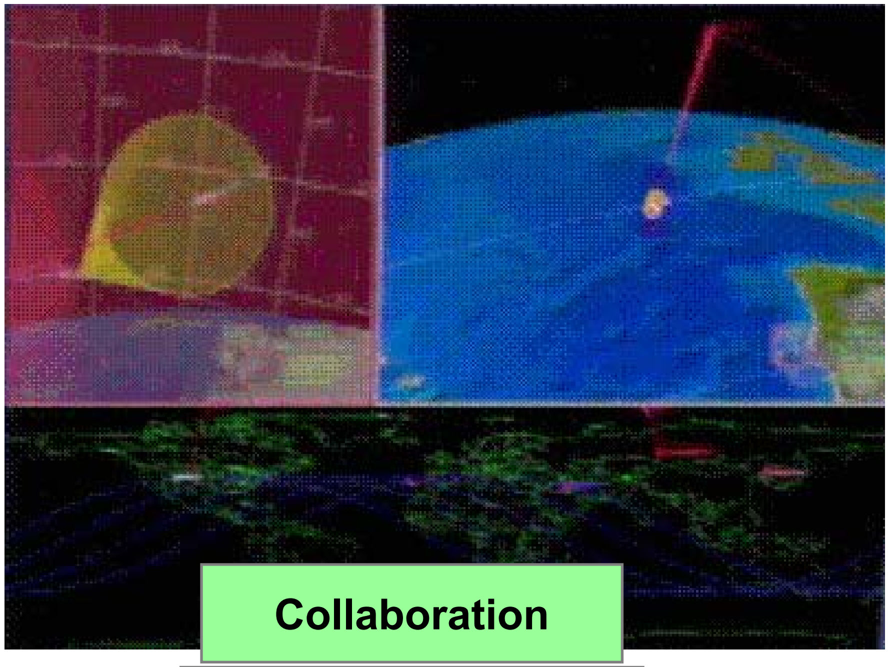
DEVS Modeling & Simulation Framework

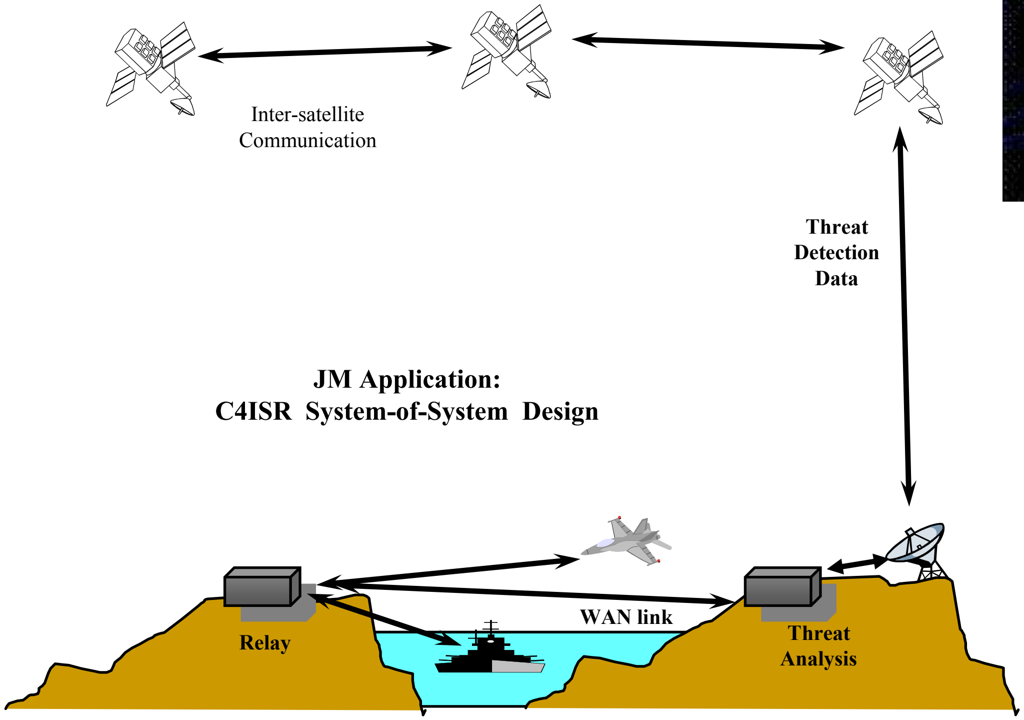
- DEVS = Discrete Event System Specification
- Provides sound M&S framework
- Derived from Mathematical dynamical system theory
- Supports hierarchical, modular composition and reuse
- Can express Discrete Time, Continuous and hybrid models
- Event-orientation enables efficient simulation

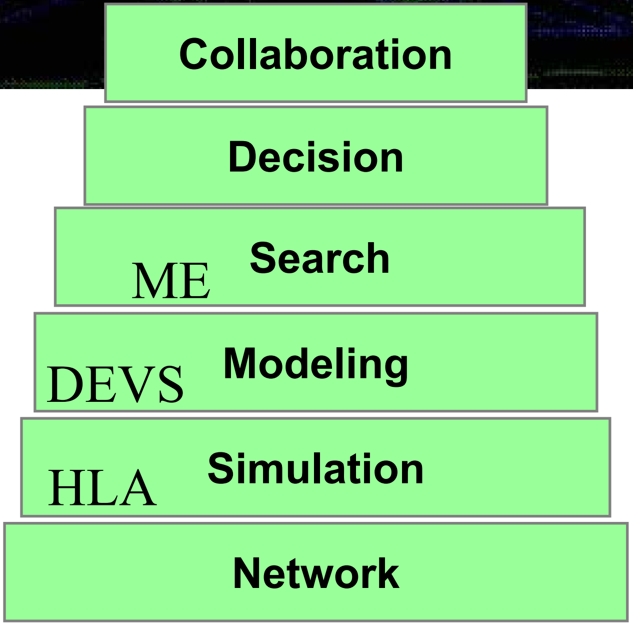
- HLA enables interoperability of existing simulations
- DEVS supports developing new simulation models within an object-oriented computational framework

Joint MEASURE™

- Jointly Developed by Lockheed and UA under DARPA ASTT
- Mission Effectiveness Simulator for System-of-Systems
- employs moderate level of resolution



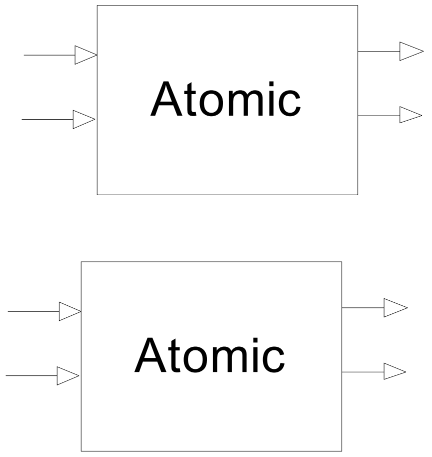

Inter-satellite Communication
Threat Detection Data
JM Application: C4ISR System-of-System Design
Relay
WAN link
Threat Analysis

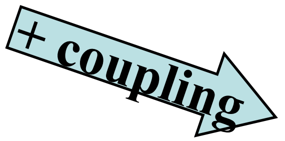


* Mission Effectiveness Analysis Simulator for Utility, Research and Evaluation

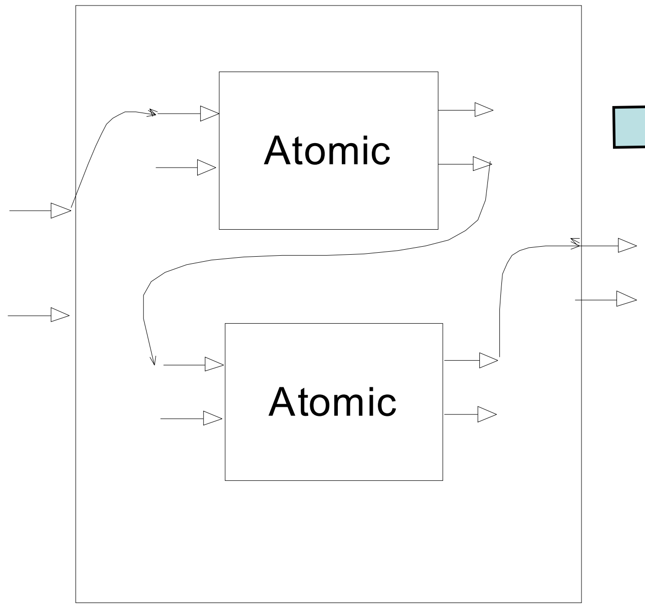
DEVS Hierarchical Modular Composition

Atomic: lowest level model, contains structural dynamics -- model level modularity

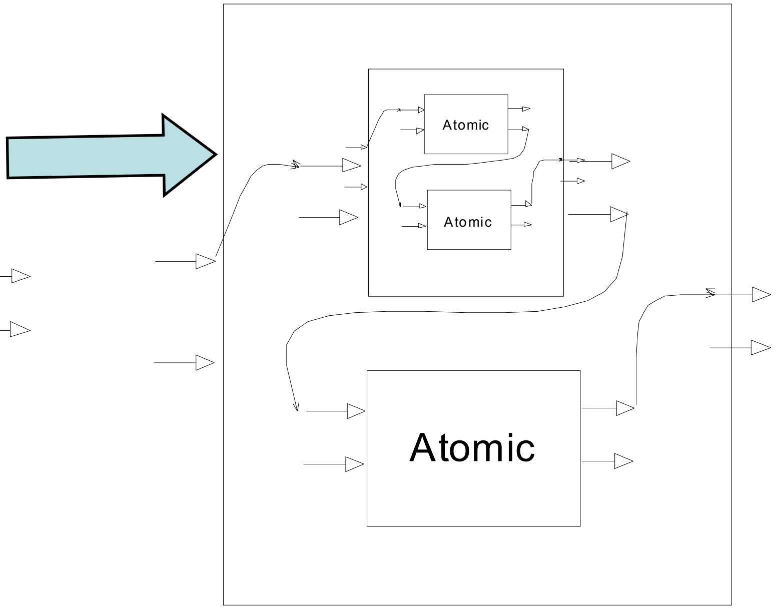




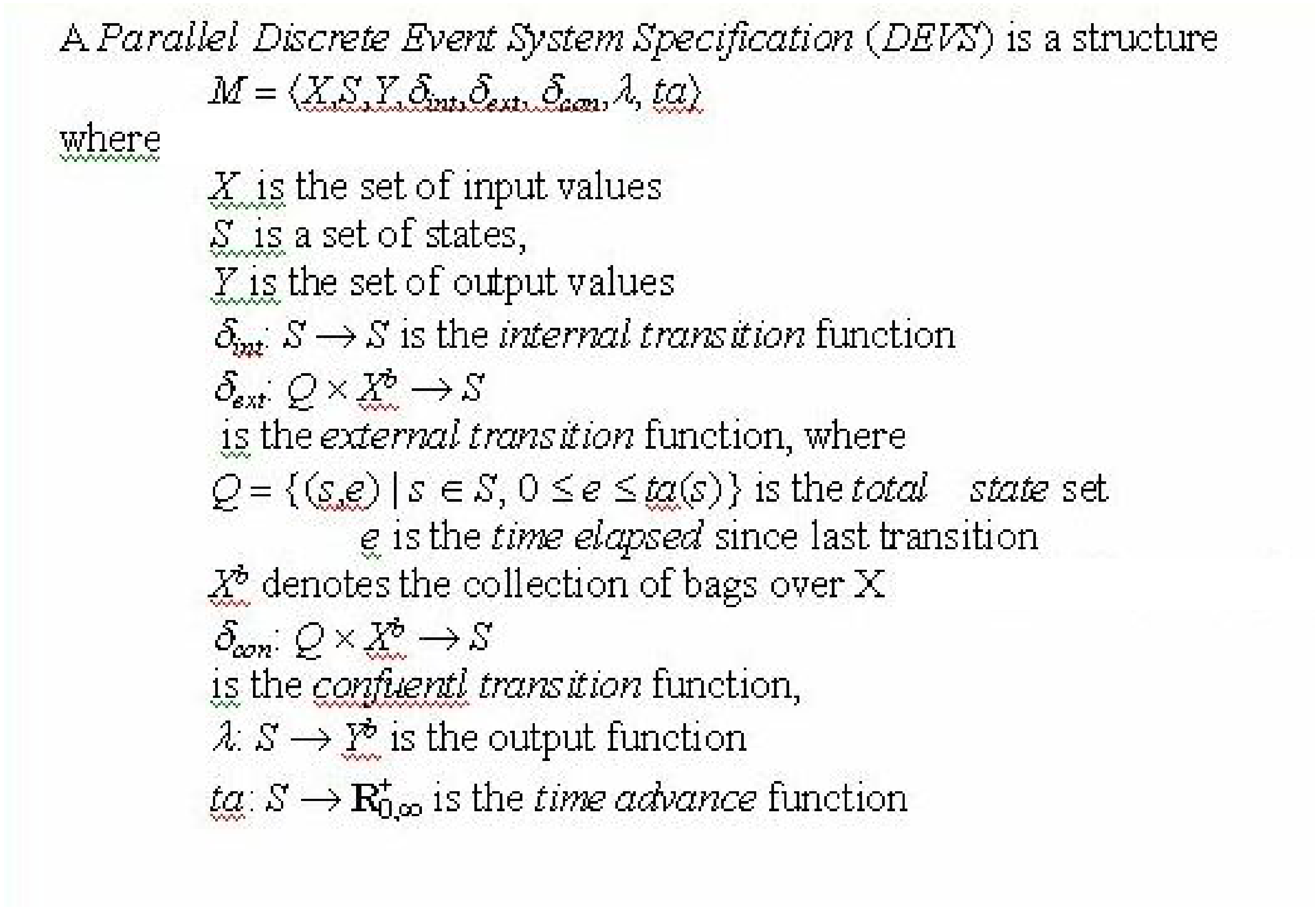
Coupled: composed of one or more atomic and/or coupled models



hierarchical construction



DEVS Formalism



A *Parallel Discrete Event System Specification* (*DEVS*) is a structure

$$M = \langle X, S, Y, \delta_{int}, \delta_{ext}, \delta_{con}, \lambda, ta \rangle$$

where

X is the set of input values

S is a set of states,

Y is the set of output values

$\delta_{int}: S \rightarrow S$ is the *internal transition* function

$\delta_{ext}: Q \times X^b \rightarrow S$

is the *external transition* function, where

$Q = \{(s,e) \mid s \in S, 0 \leq e \leq ta(s)\}$ is the *total state* set

e is the *time elapsed* since last transition

X^b denotes the collection of bags over X

$\delta_{con}: Q \times X^b \rightarrow S$

is the *confluentl transition* function,

$\lambda: S \rightarrow Y^b$ is the output function

$ta: S \rightarrow \mathbf{R}^{+}_{0,\infty}$ is the *time advance* function

Coupled Model Specification

$$\mathbf{DN} = < \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{D}, \{\boldsymbol{M}_i\}, \{\boldsymbol{I}_i\}, \{\boldsymbol{Z}_{i,j}\} >$$

$\boldsymbol{X}$: a set of input events.

$\boldsymbol{Y}$: a set of output events.

$\boldsymbol{D}$: an index set (names) for the components of the coupled model.

For each $\boldsymbol{i} \in \boldsymbol{D}$,

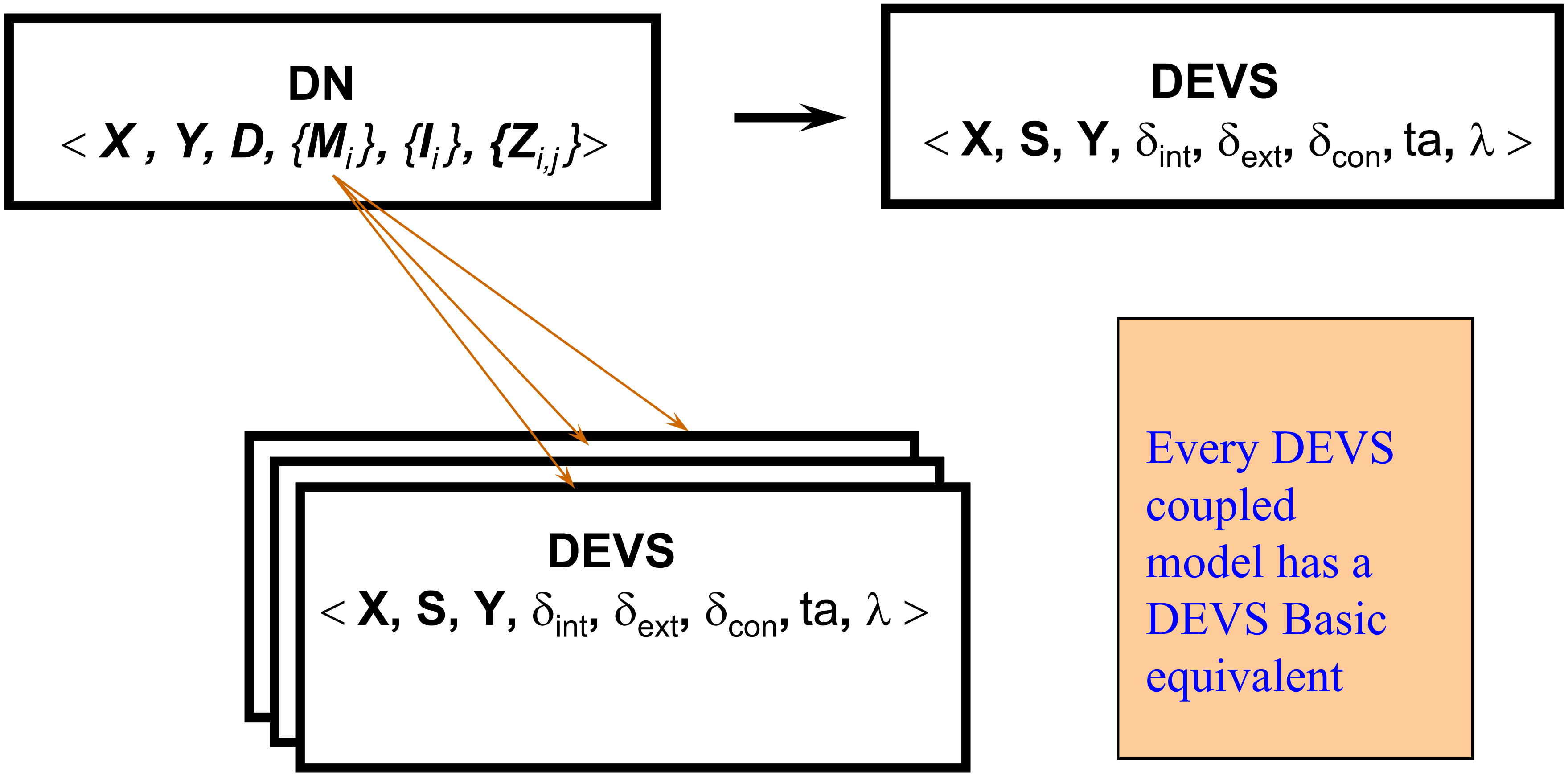
$\boldsymbol{M}_i$ is a component DEVS model.

For each $\boldsymbol{i} \in \boldsymbol{D} \cup \boldsymbol{self}$, $\boldsymbol{I}_i$ is the set of influencees of $\boldsymbol{i}$.

For each $\boldsymbol{j} \in \boldsymbol{D} \cup \boldsymbol{self}$,

$\boldsymbol{Z}_{i,j} : \boldsymbol{Y}_i \rightarrow \boldsymbol{X}_j$ is the output translation mapping

Closure Under Coupling


DN
$< \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{D}, \{\boldsymbol{M}_i\}, \{\boldsymbol{I}_i\}, \{\boldsymbol{Z}_{i,j}\}>$
DEVS
$< \mathbf{X}, \mathbf{S}, \mathbf{Y}, \delta_{int}, \delta_{ext}, \delta_{con}, ta, \lambda >$
DEVS
$< \mathbf{X}, \mathbf{S}, \mathbf{Y}, \delta_{int}, \delta_{ext}, \delta_{con}, ta, \lambda >$

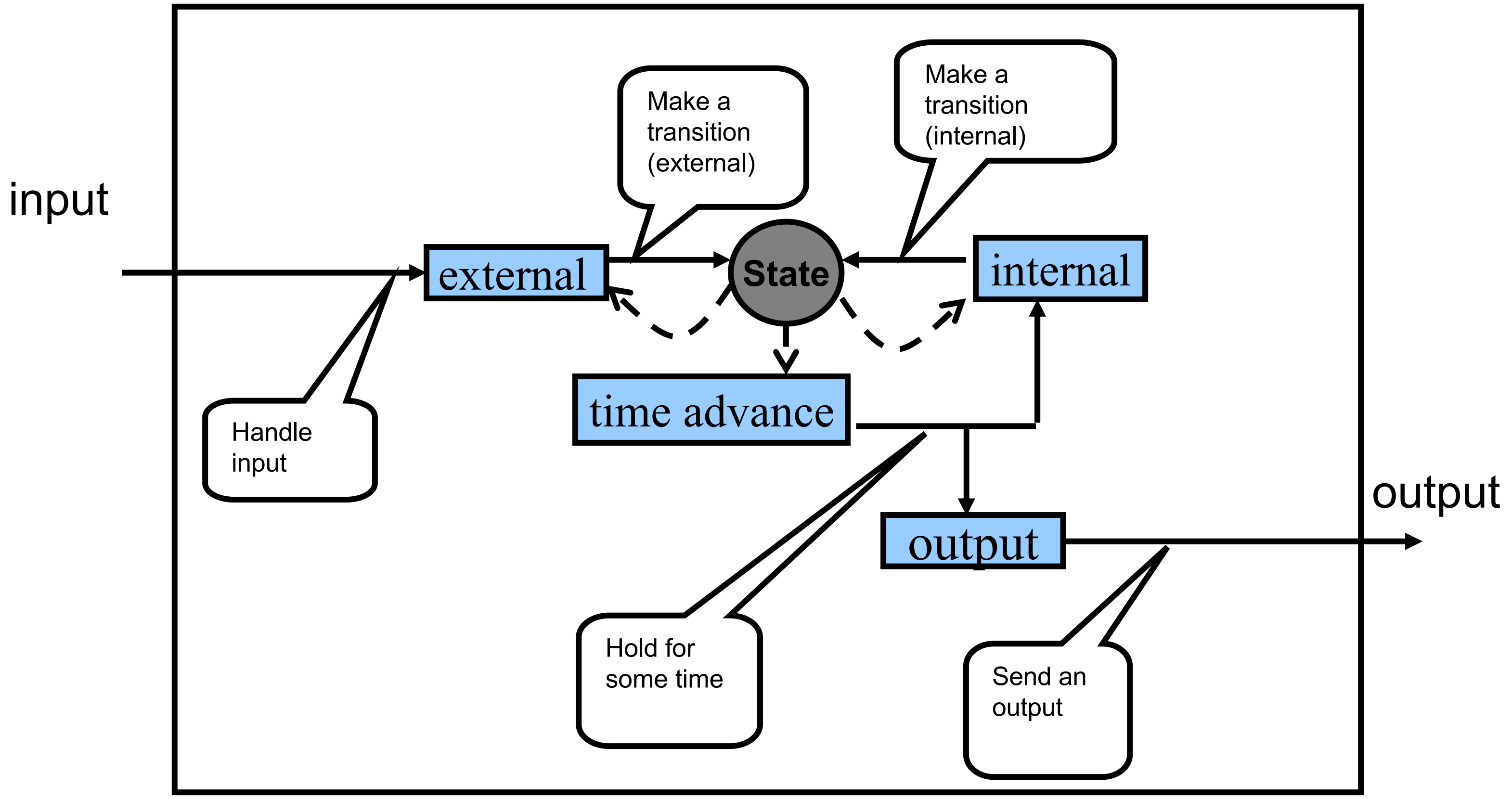
Every DEVS coupled model has a DEVS Basic equivalent

DEVS Atomic Model

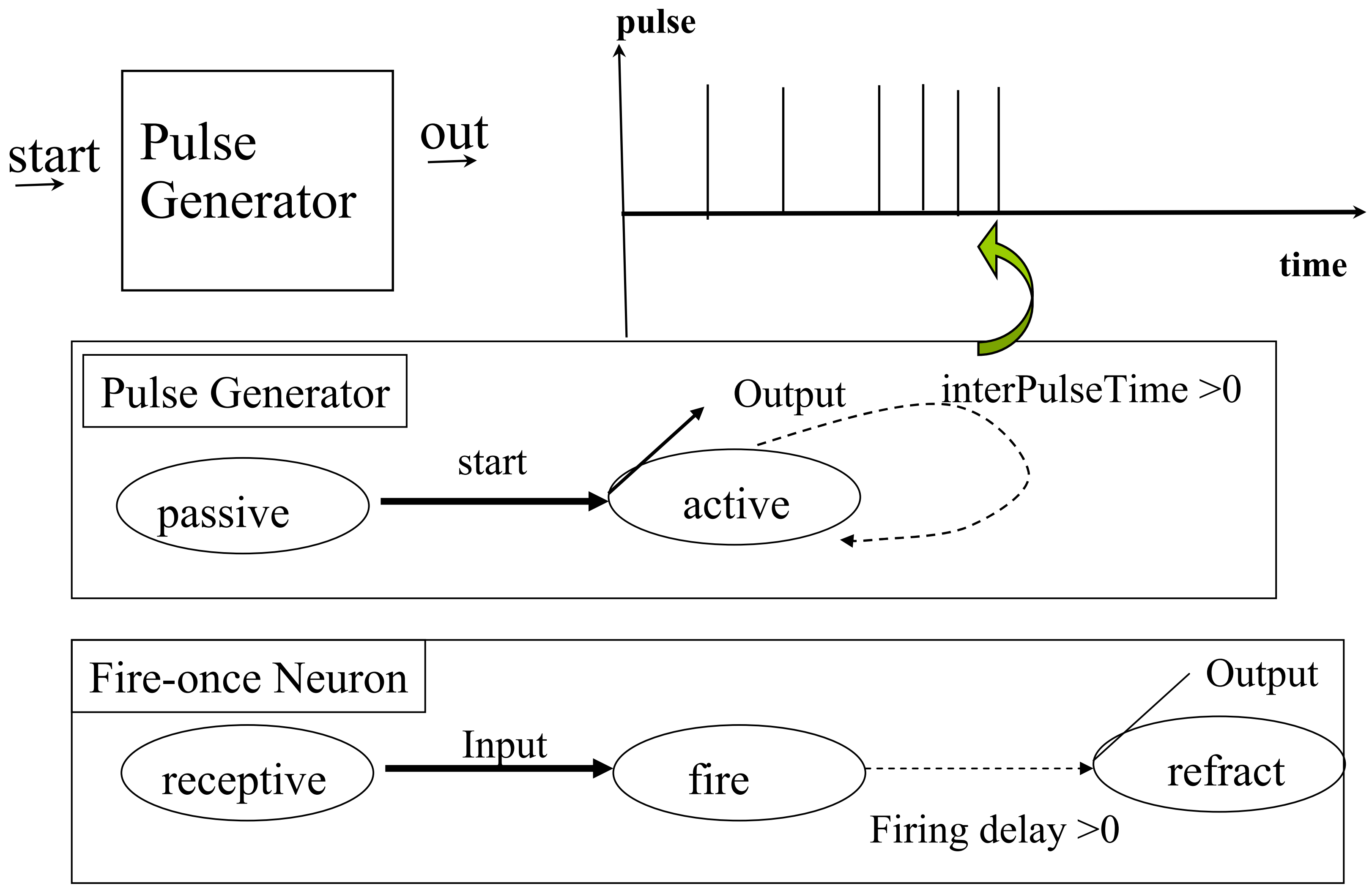
Elements of an atomic model:

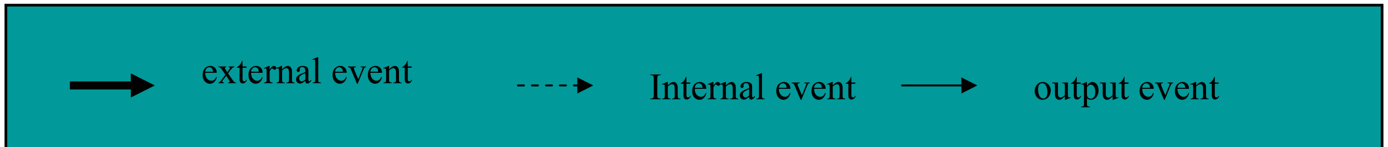
- input events
- output events
- state variables
- state transition functions
- output function
- time advance function

DEVS Atomic Model
Implements Basic DEVS

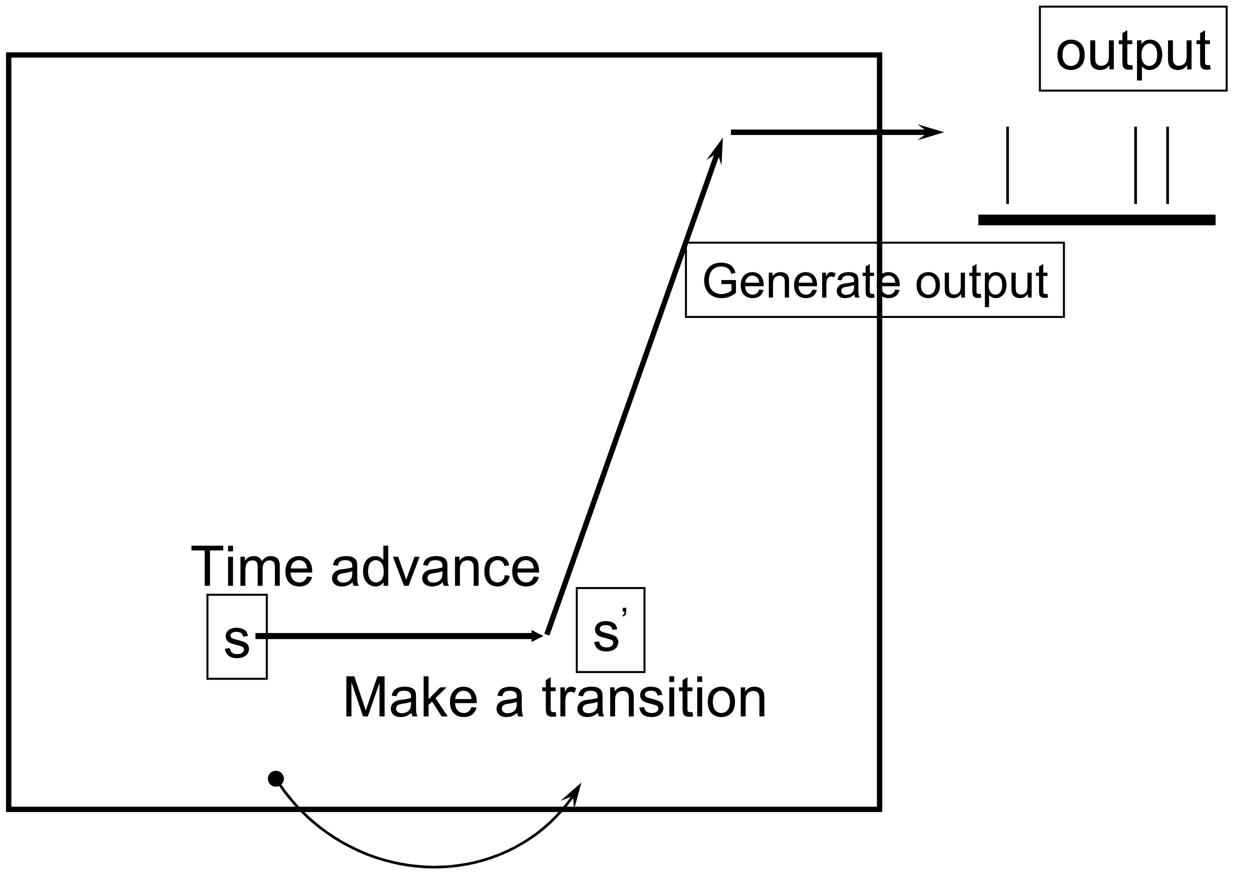


Atomic Model Examples

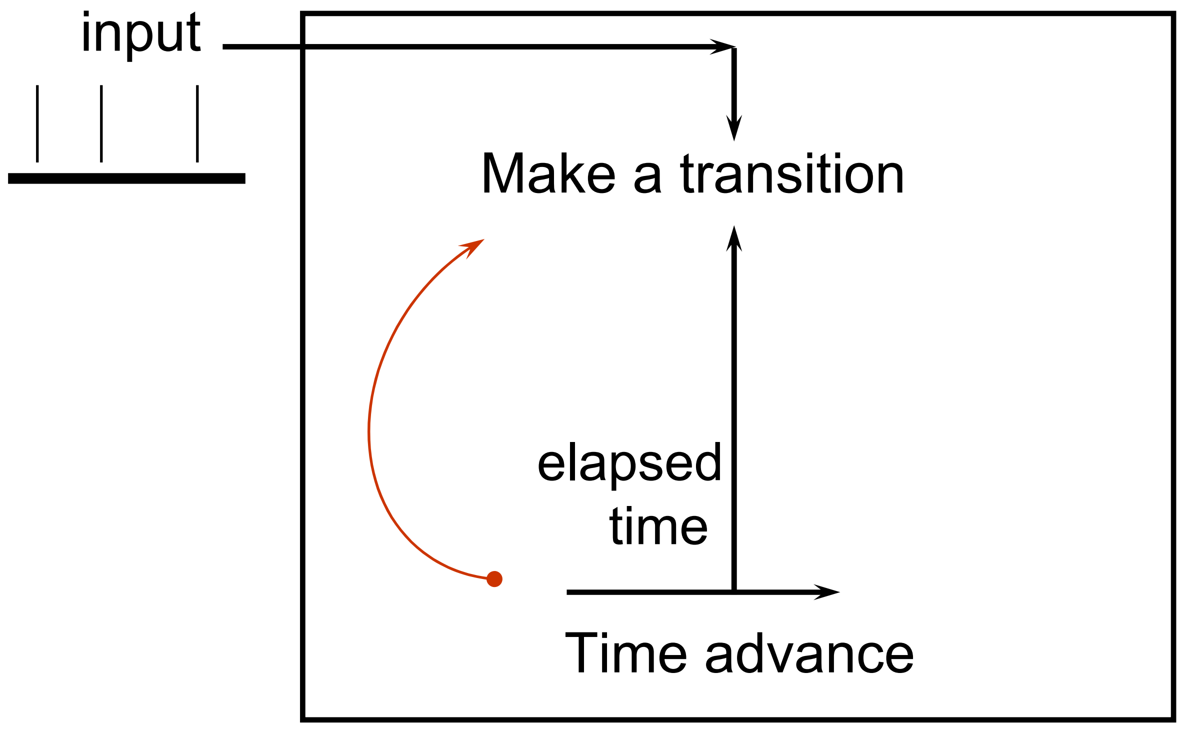




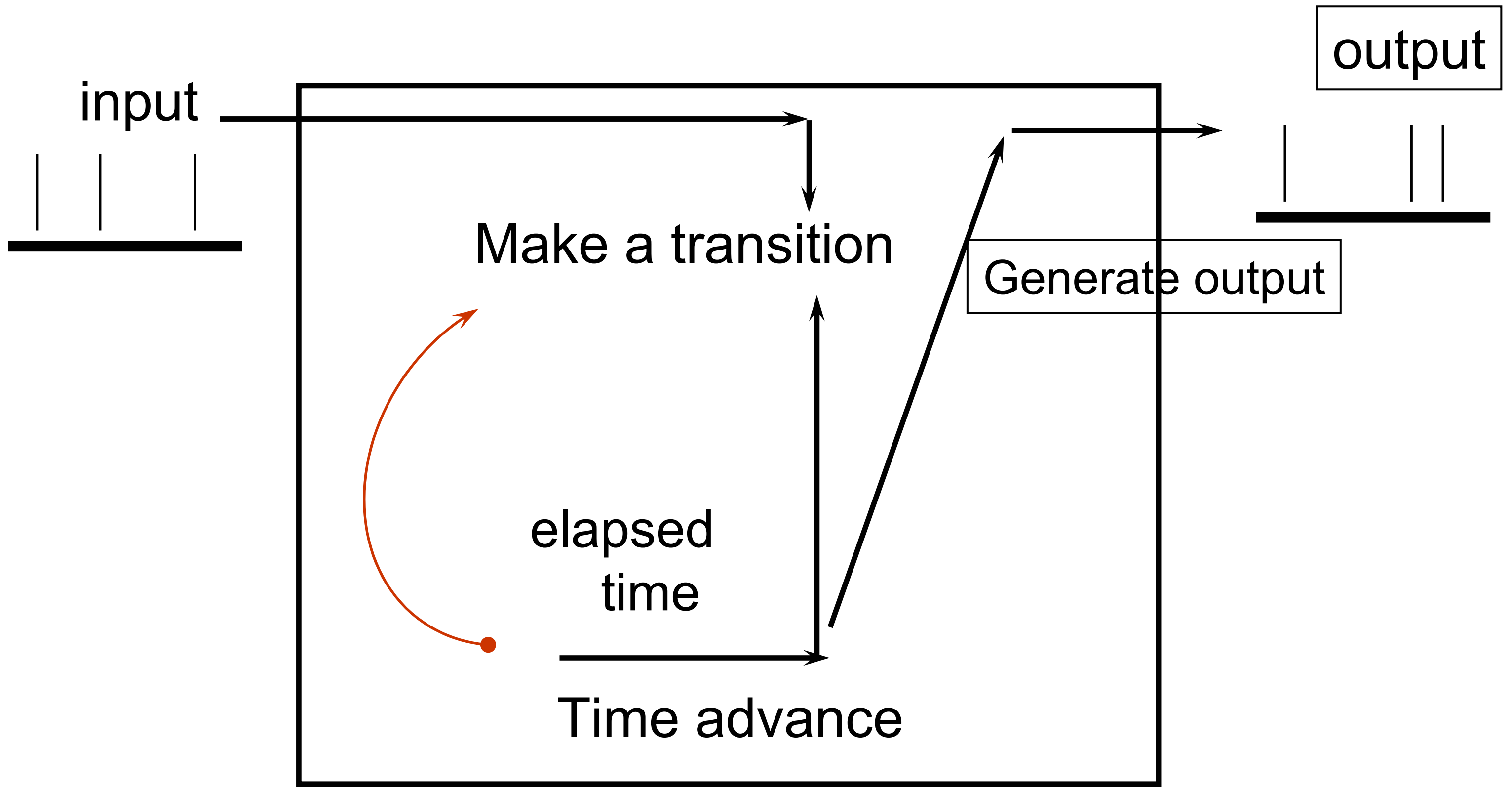
Internal Transition /Output Generation



Response to External Input



Response to Simultaneous External Input and Internal Event

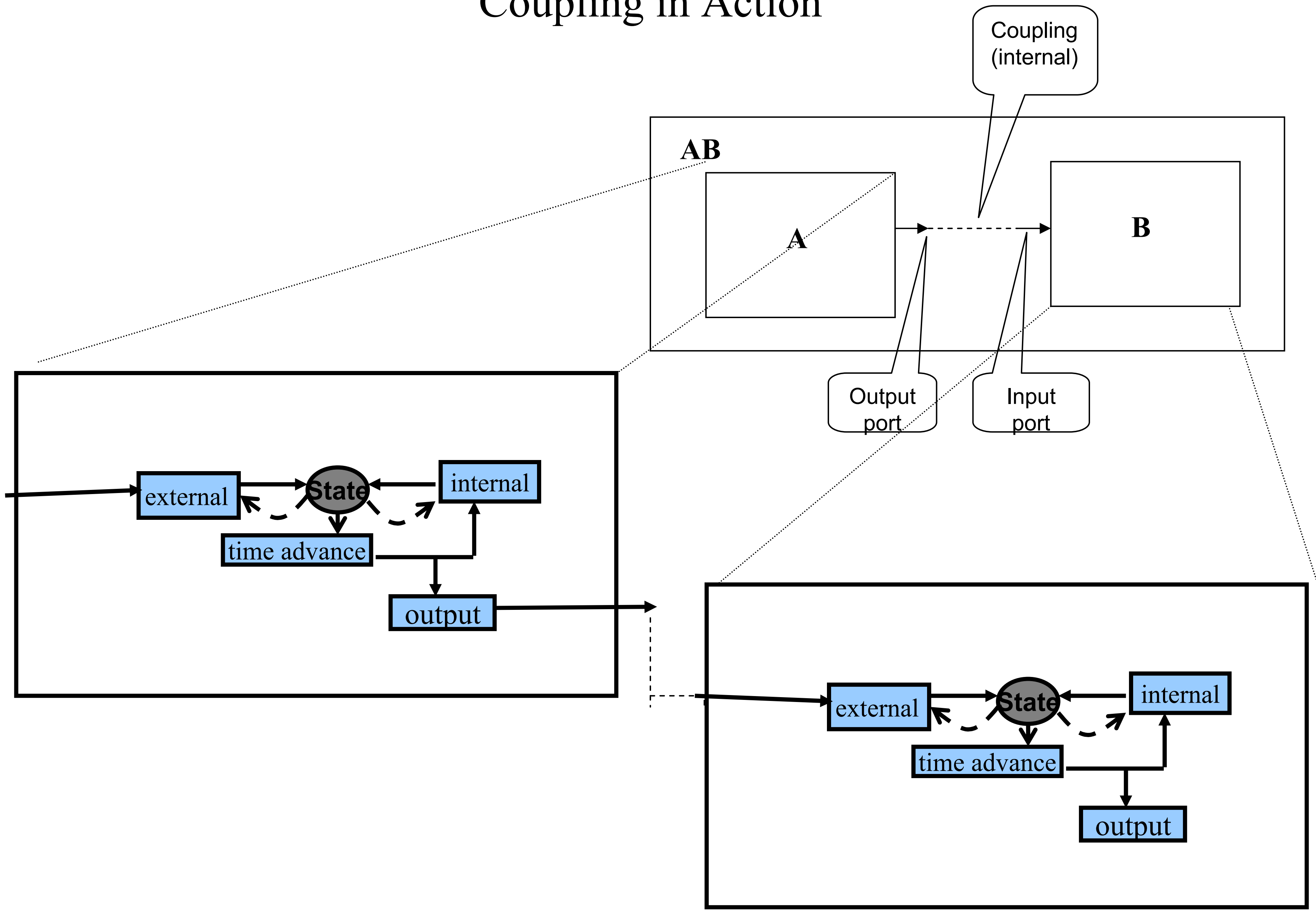


DEVS Coupled Model

Elements of coupled model:

- Components
- Interconnections
  - Internal Couplings
  - External Input Couplings
  - External Output Couplings

Coupling in Action

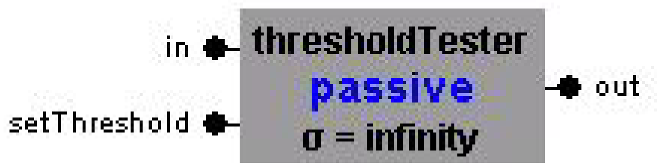


Experimental Frame Components

An experimental frame specifies the conditions under which a model or real system is experimented or tested

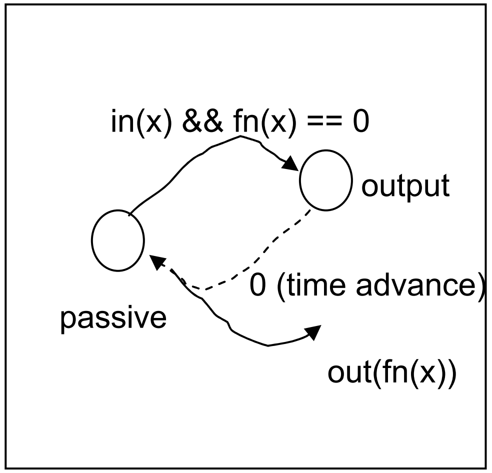
- Some useful components are:
- threshold tester
- timer
- stopwatch

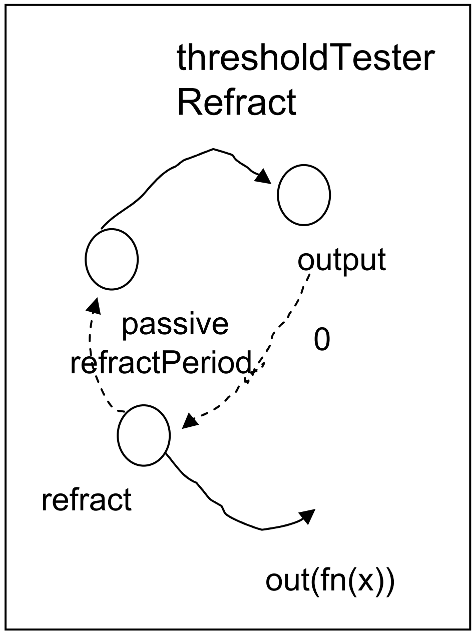
class threshold tester – tests any incoming real value for crossing of threshold

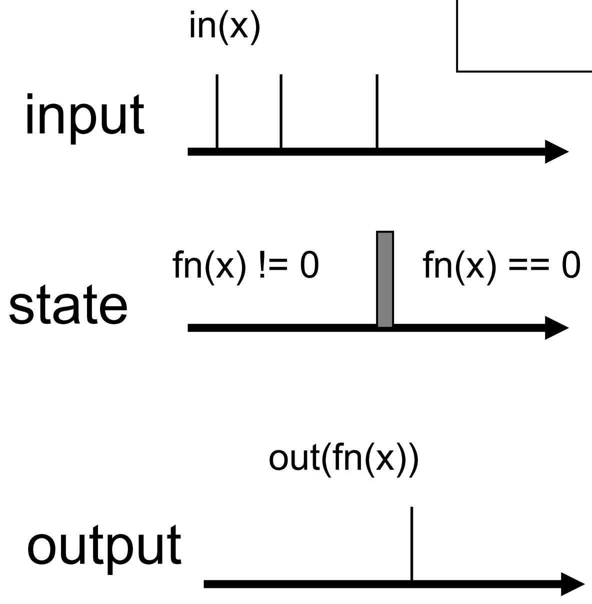


```
public double fn(double x){
if (x >= threshold)
return 0;
else return 1;
}
```

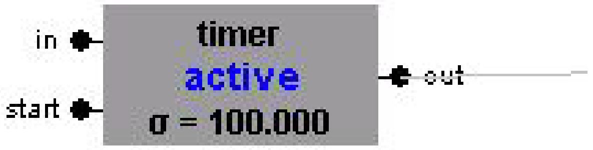
```
public void deltext(double e,message x){
if (somethingOnPort(x,"setThreshold"))
threshold =
getRealValueOnPort(x,"setThreshold");

if (somethingOnPort(x,"in")){
inval = getRealValueOnPort(x,"in");
outval = fn(inval);
if (outval == 0)
holdIn("output",0);  //only output a 0 if threshold
passed
}
else passivate();
}
```



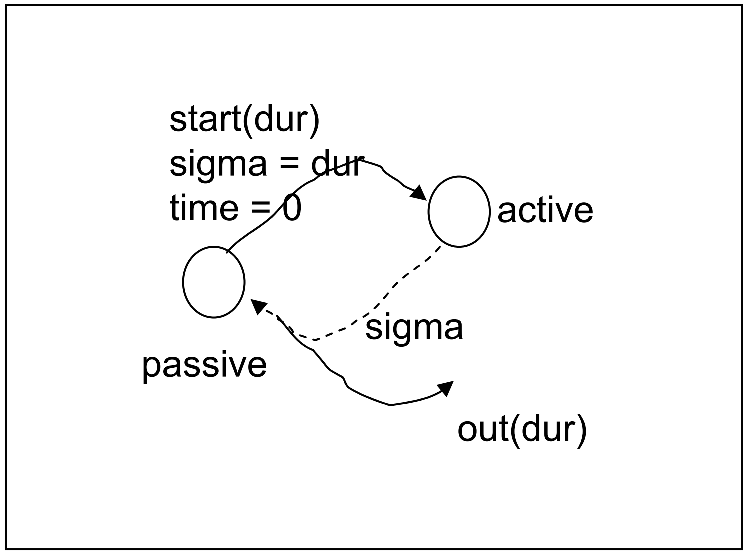


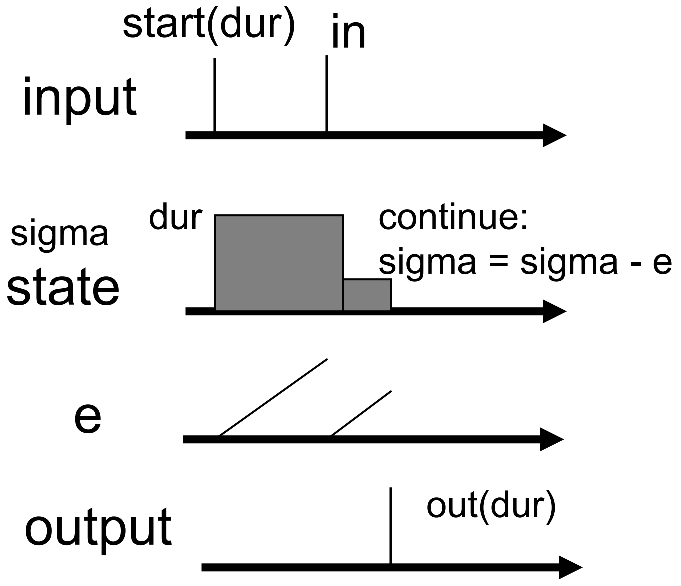


class timer – waits for a specified time then outputs pulse

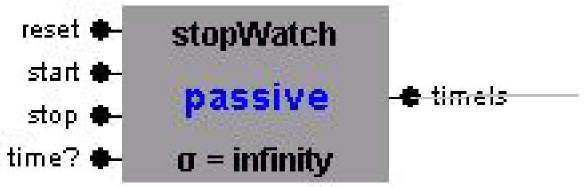


```
public void deltext(double e, message x)
  {
     Continue(e);
     time += e;
   if (phaseIs("passive"))
    if (somethingOnPort(x,"start")){
      double dur = getRealValueOnPort(x,"start");
       time = 0;
       holdIn("active",dur);
    }
}

public void deltint()
  {
       time += sigma;
       passivate();
  }

public message out()
  {
   if (phaseIs("active"))
   return outputRealOnPort(time+sigma,"out");
   else return outputRealOnPort(0,"dum");
  }
```

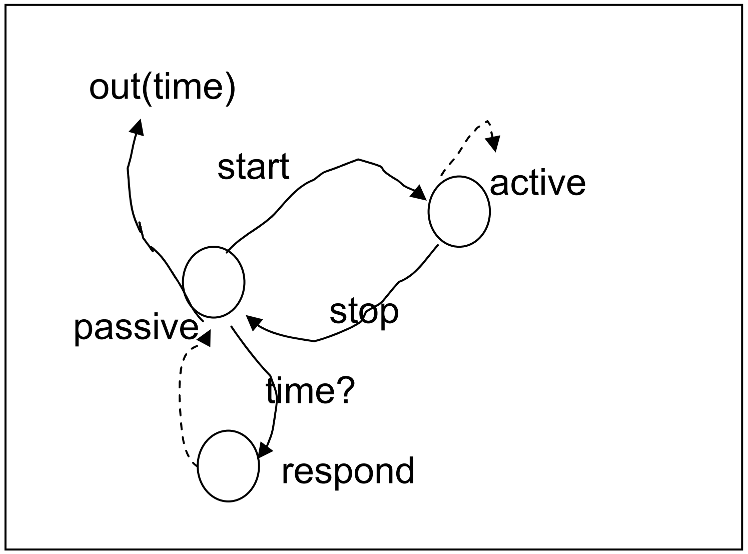


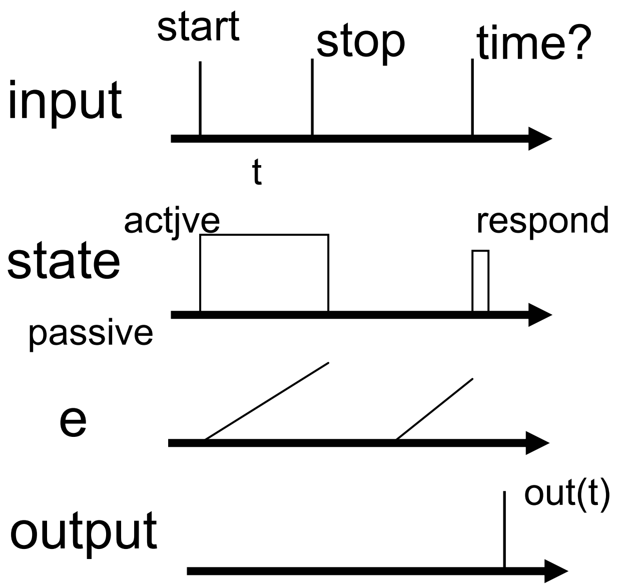


class stopwatch – measures and reports elapsed time



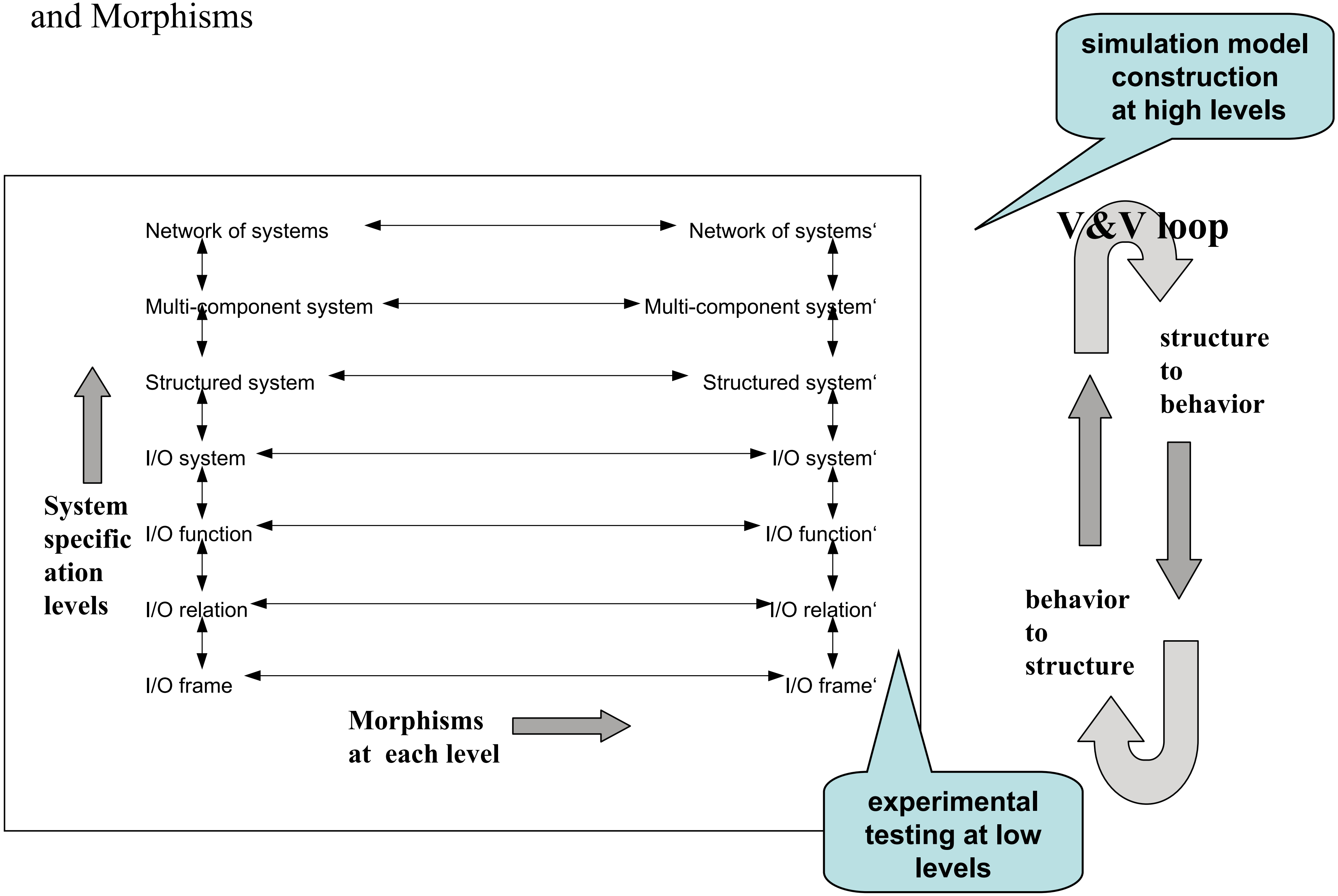
```
public void deltext(double e, message x)
  {
     Continue(e);
  if (somethingOnPort(x,"time?")){
    if (phaseIs("passive"))
     response = true;  //for simultaneous stop & reset
    else passivate();
  }
  else if (somethingOnPort(x,"start"))
  passivateIn("active");
  else if (somethingOnPort(x,"stop")){
  time += e;
  passivate();
  }
  else if (somethingOnPort(x,"reset"))
  time = 0;

  if (response) holdIn("respond", 0);
  }

  public message out()
  {
  if (phaseIs("respond"))
  return outputRealOnPort(time,"timeIs");
  else
  return outputNameOnPort("","dum");
  }
```



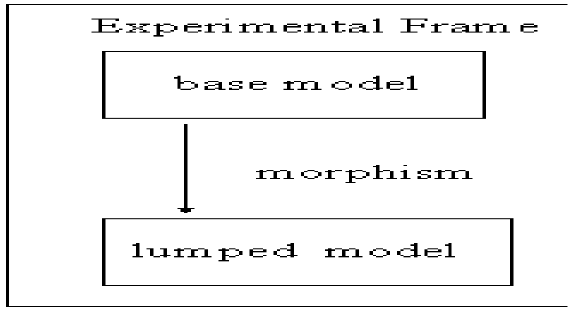


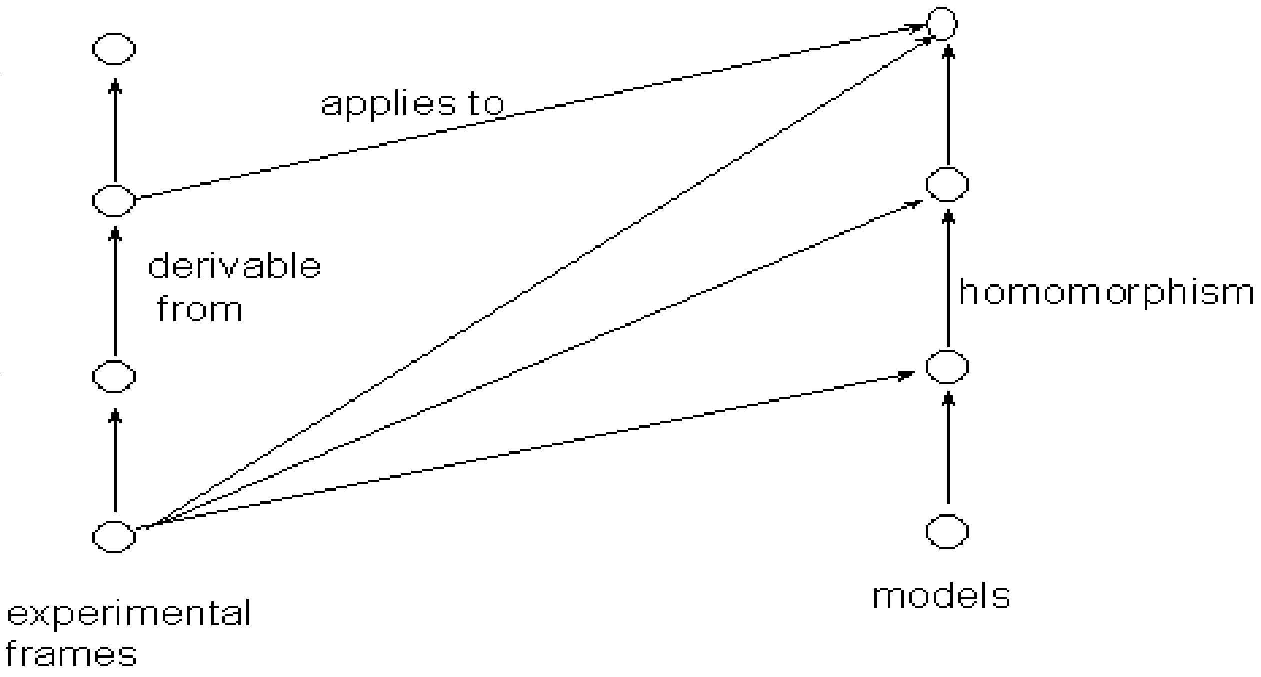
M&S/System Theory Relationships in the context of V&V

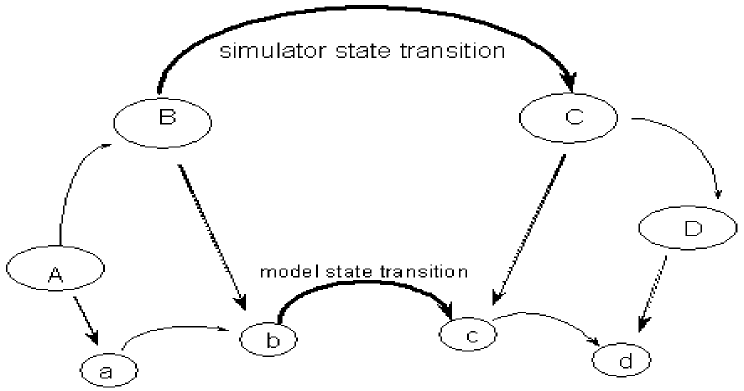
Hierarchy of System Specifications and Morphisms



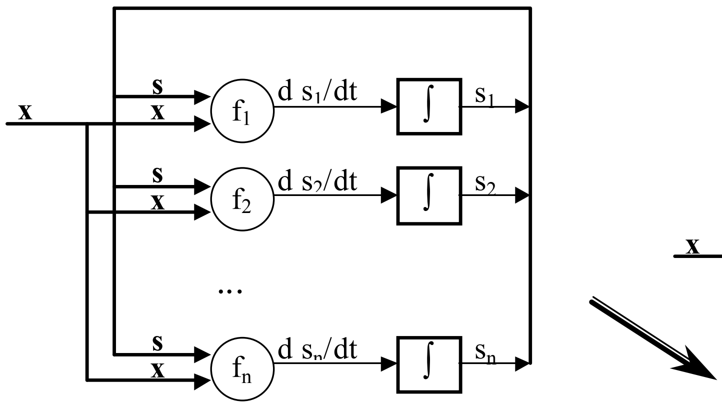
M&S Relations (cont'd)

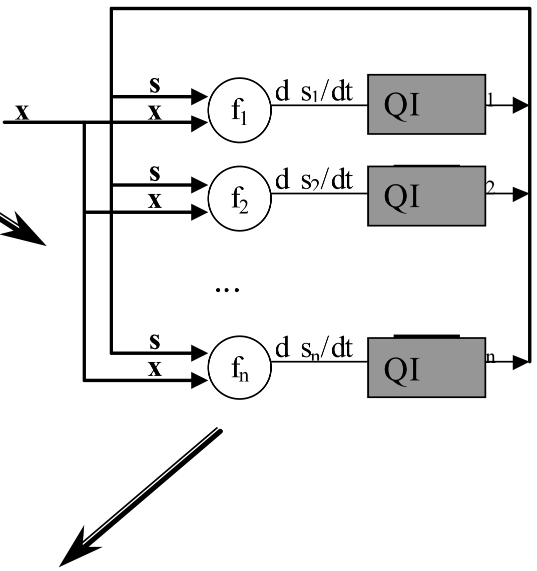


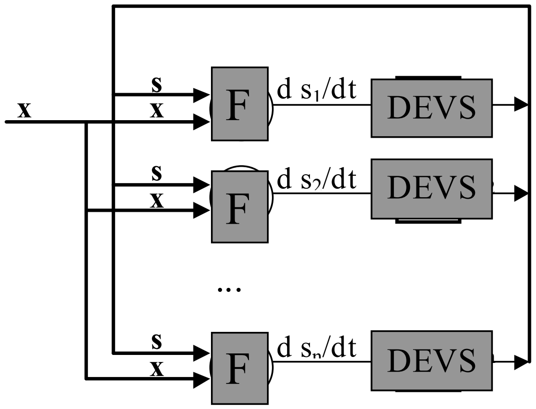




Differential Equation to DEVS-- Approximate Component-wise Morphism







Each source component is mapped into its target representative

This creates a network-of-systems morphism which is approximate

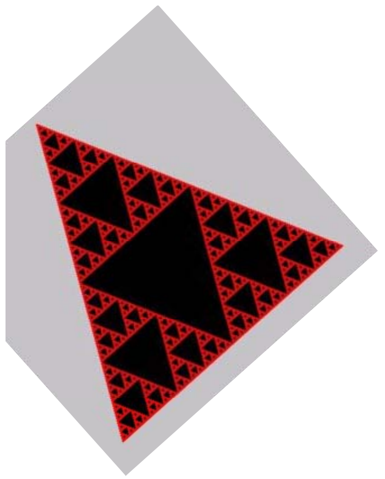
Example: Some Cellular Spaces

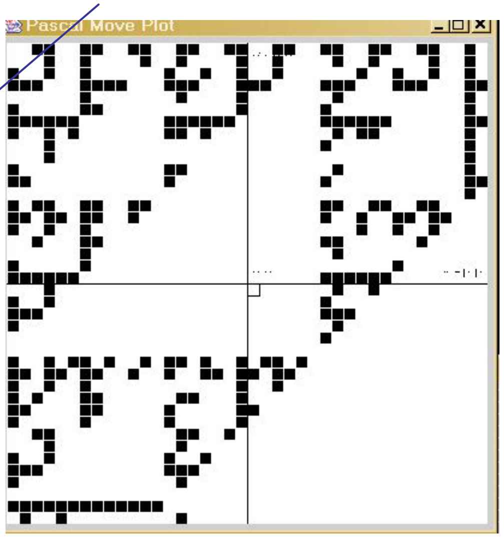
Cellular Automaton Realization of Pascal’s Triangle

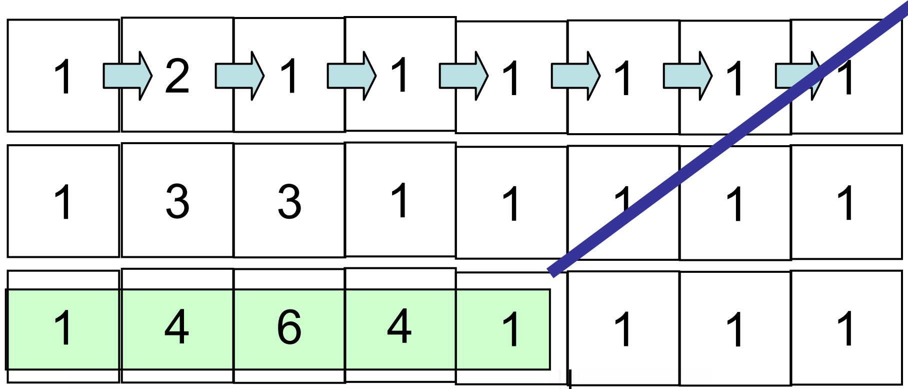
Implementation:
- A one-dimensional cell space with left to right neighborhood such that each cell adds its neighbors sate to its own at each time step.
- The exceptions are the initial passive cells o the right of the “2”.
- They are activated as the non-unity activity reaches them.
- Row counts are incremented at each step for active cells
- Their states are plotted at points (cell index, cell row) using parity coding (modulo 2).
- The resulting pattern turns the triangle on its side.

The first seven rows of Pascal's Triangle look like:

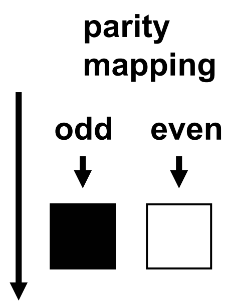
```
                1                       n=0
             1     1                    n=1
          1     2     1                 n=2
       1     3     3     1              n=3
    1     4     6     4     1           n=4
 1     5    10    10     5     1        n=5
1   6    15    20    15    6     1      n=6
```

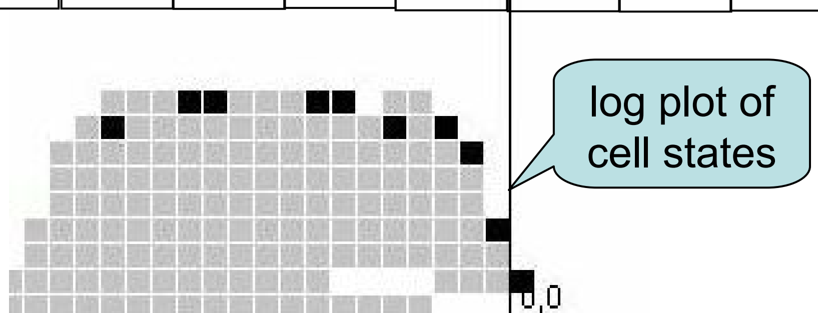






1	2	1	1	1	1	1	1
1	3	3	1	1	1	1	1
1	4	6	4	1	1	1	1



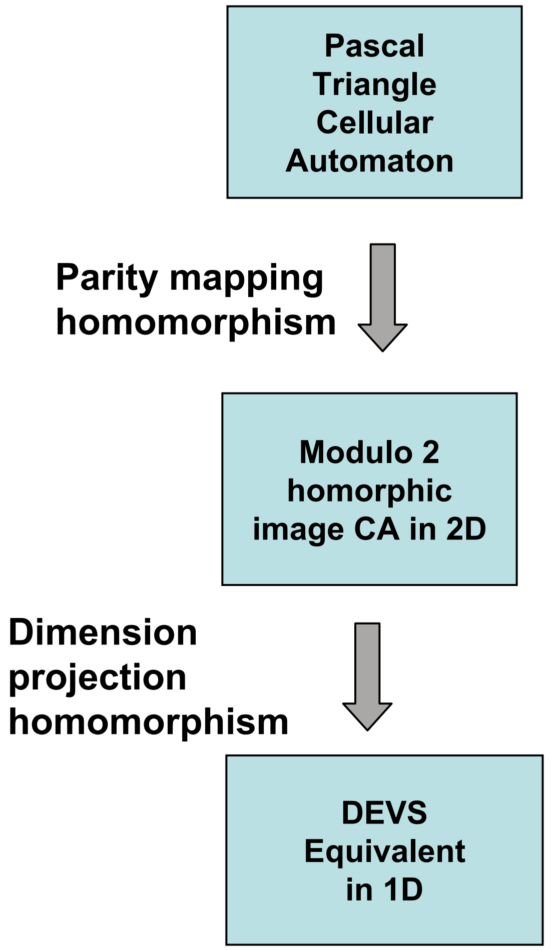


Game, set, and math : enigmas and conundrums / Ian *Stewart.* Oxford [England] ; Cambridge, Mass, USA : B. Blackwell, 1989.

http://www.cs.washington.edu/homes/jbaer/classes/blaise/blaise.html

A New Kind of Science, Wolfram

Base and Lumped Models and Morphism For Pascal’s Triangle

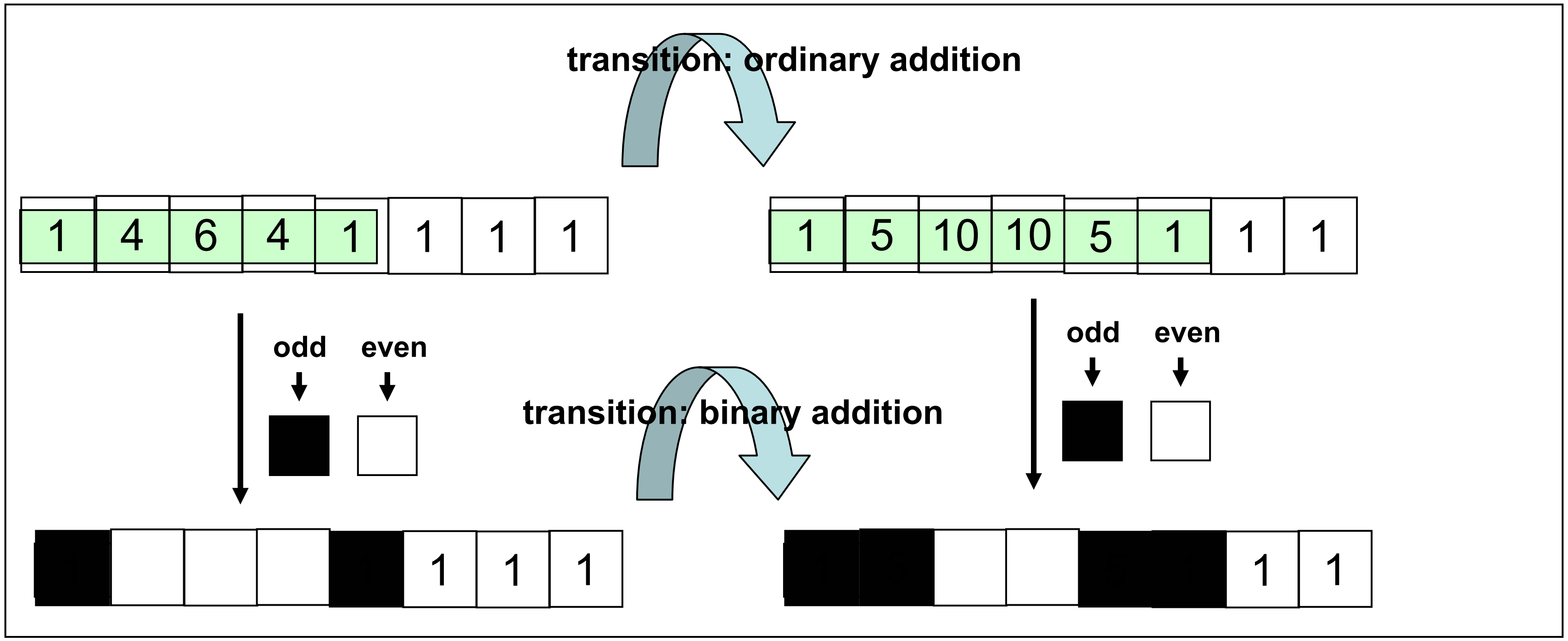

Pascal Triangle Cellular Automaton
Parity mapping homomorphism
Modulo 2 homorphic image CA in 2D
Dimension projection homomorphism
DEVS Equivalent in 1D

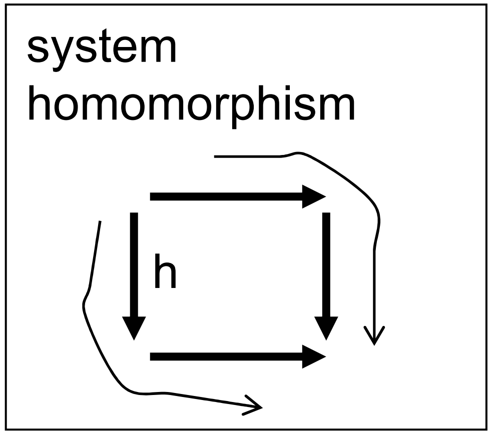
- Base Model

- Binary cell states represent parity of parent cell values
- Binary addition replaces integer addition
for local transitions

- Cells output only when a true state change occurs
- They store last values of inputs, using these until updated
by neighbor messages
- They passivate if no state change occurs, to be revived
when a new input arrives.

The Cell-wise Parity mapping is a system homomorphism





even + even = even
odd + odd = even
odd + even = odd
even + odd = even

This system has an analytic solution – the color can be predicted from a cells coordinates without simulation – this is not true in general

A mapping from base model to lumped model state spaces is a homomorphism if it commutes with the transition and output functions

Satisfaction of this criterion allows the lumped model to replace the base model in the relevant experimental frame without error

Here this means we can use binary addition in the transition function and generate the Pascal Triangle starting from the initial binary state (101111…).

Model-Experimental Frame Relationships for Pascal Triangle Domain

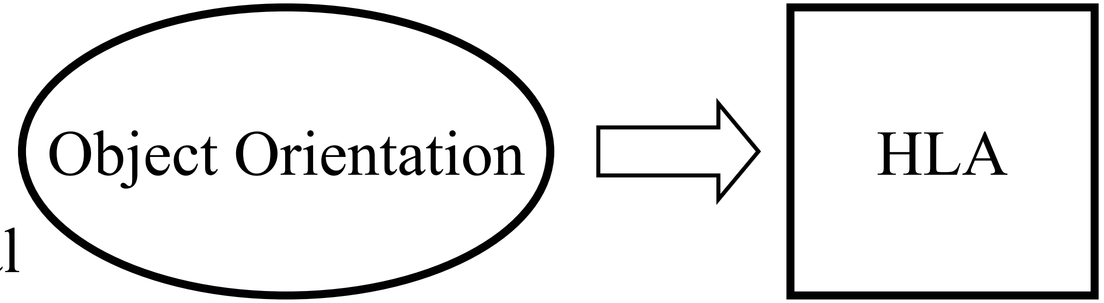
derivability →

↓ **morphism**

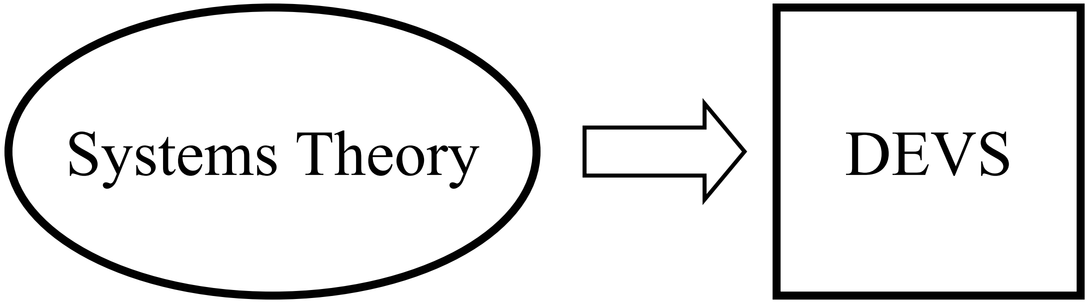
Output / Model	Binary Coefficients	Black-white Pattern In 2D	Black-white Pattern In 1D	Speed
Pascal Triangle CA	√	√	√	**Unlimited size of operand**
Modulo 2 Homomorphic Image CA in 2D		√	√	**Faster due to limits size of operands**
DEVS-equivalent Mod 2 image in 1D			√	**Faster because based on events --- sparcity increases with time, i.e., asymptotically approaches almost all white**

Systems Theory and Object Orientation

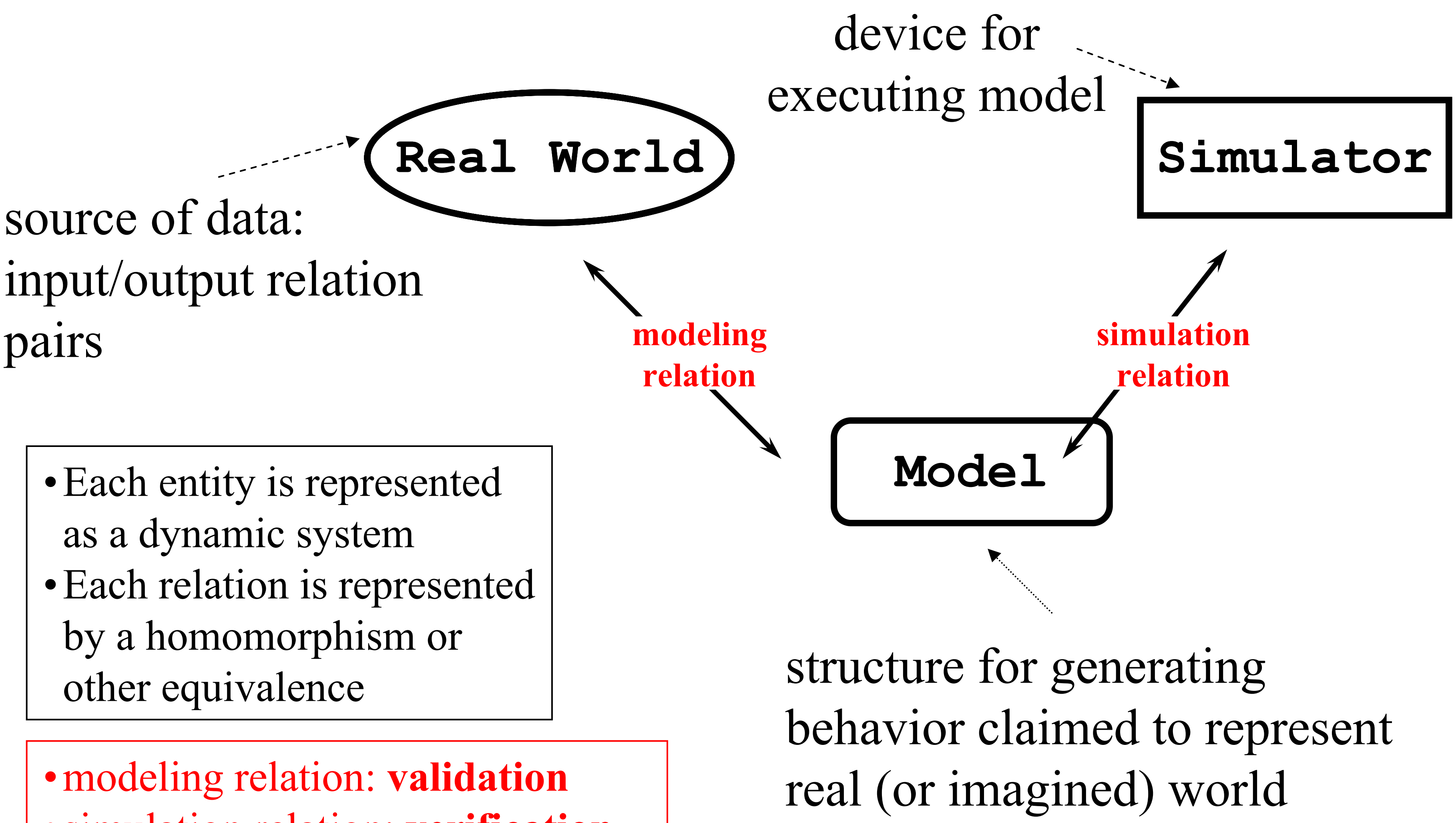
- Software/system development (handles primarily computational aspects)



- System characterization and requirements (handles primarily modeling aspects)



M&S Entities and Relations

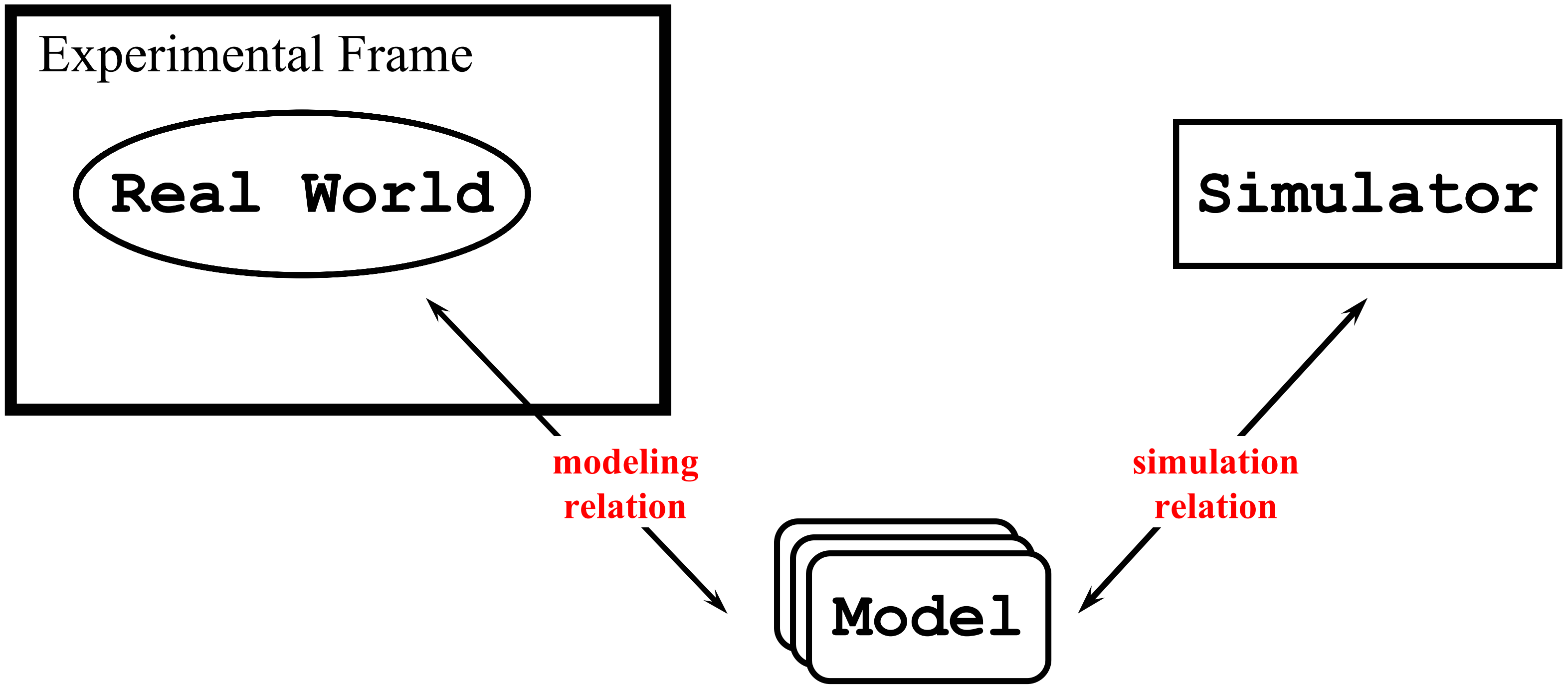


device for executing model
Real World
Simulator
source of data: input/output relation pairs
modeling relation
simulation relation
Model
structure for generating behavior claimed to represent real (or imagined) world

- Each entity is represented as a dynamic system
- Each relation is represented by a homomorphism or other equivalence

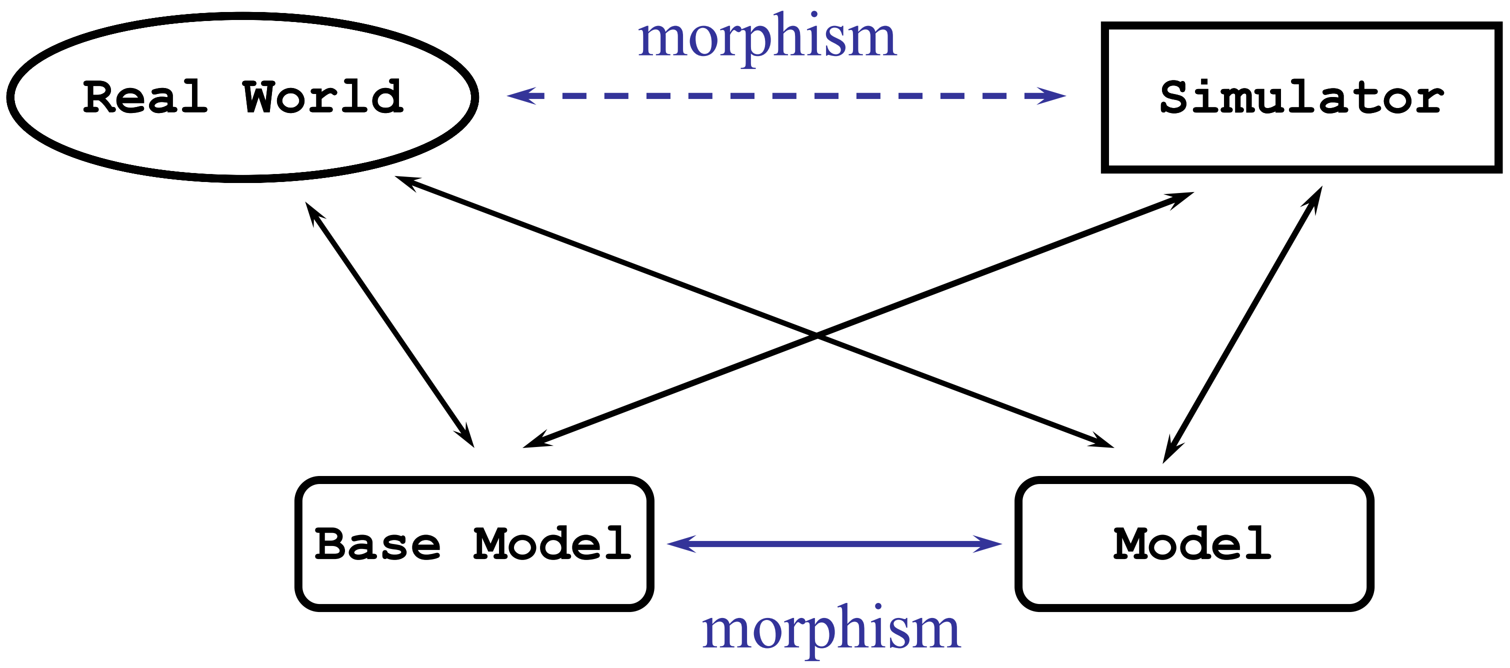
- modeling relation: **validation**
- simulation relation: **verification**

M&S Entities and Relation(cont.)



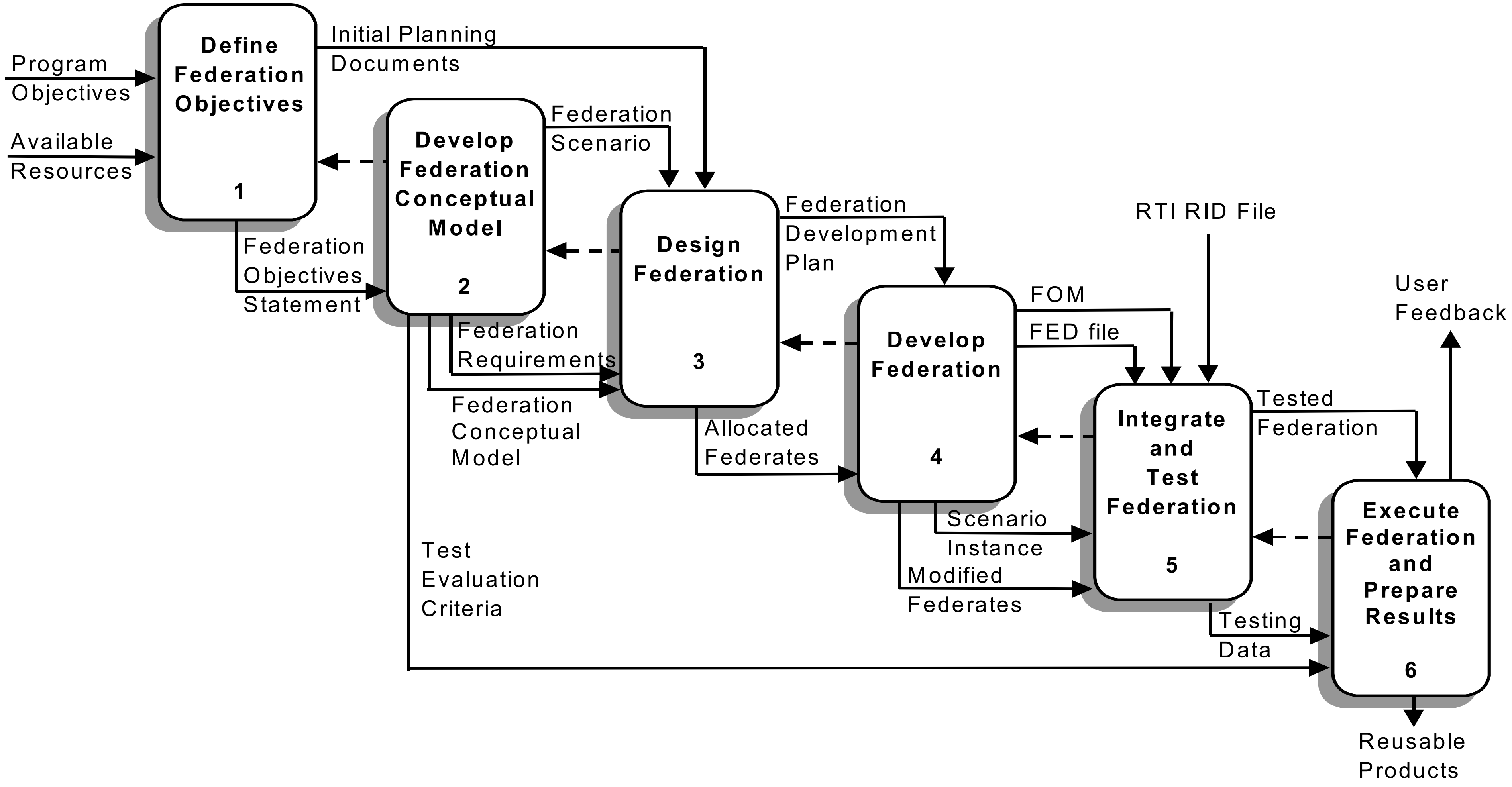
- Experimental frame specifies conditions under which the system is experimented with and observed

M&S Entities and Relation(cont'd)



- Base model characterizes real world in greater details (inputs, outputs, states, and functionality) compare to a (lumped) model – the variety of behavior it can generate supersedes those of its corresponding lumped model
- Lumped model is often sufficient given complexity and cost associated with its base model

FEDEP Model

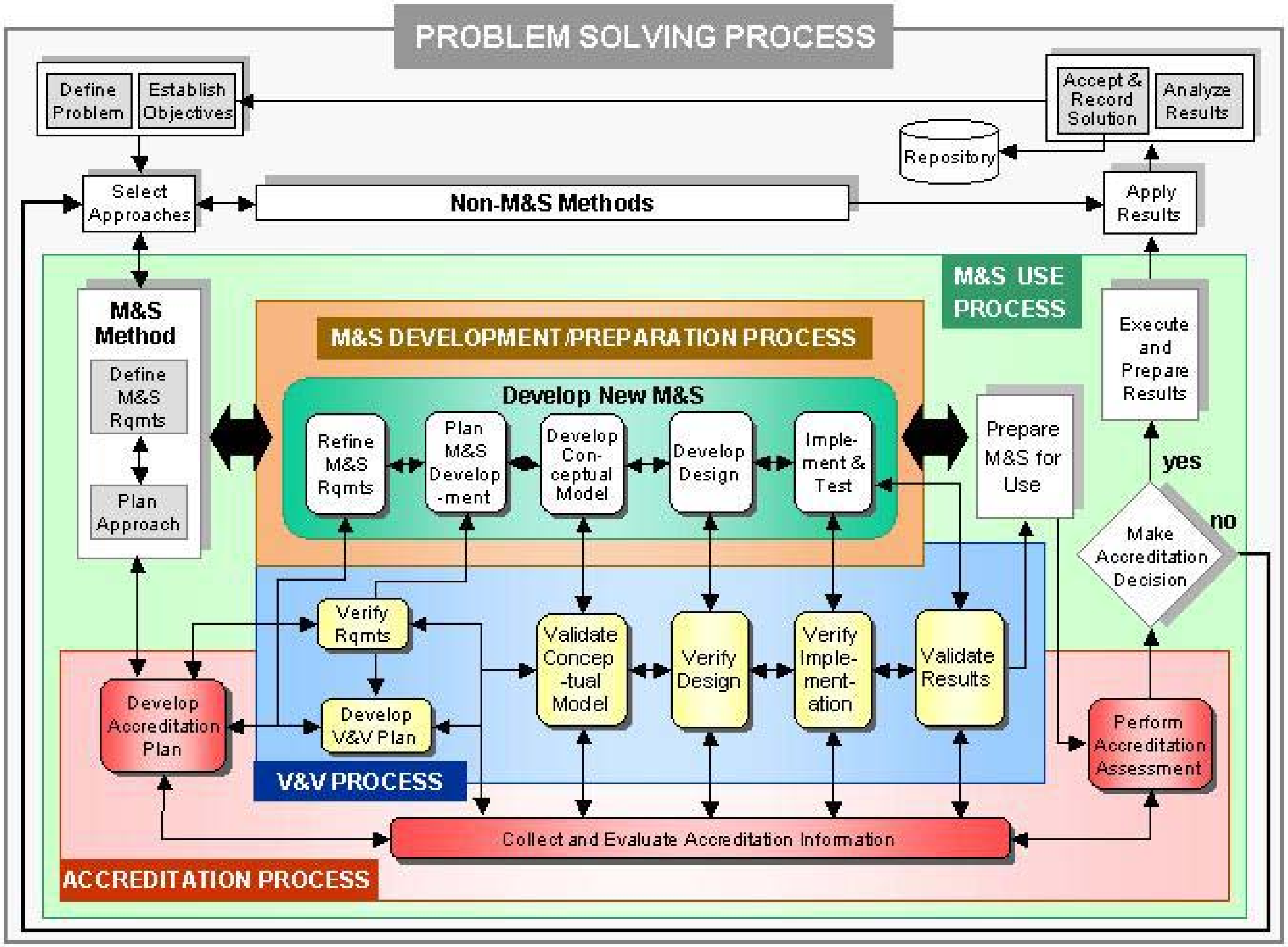


R.C. Turrell, et al. (99s-SIW-130.ppt)

Limitations of FEDEP

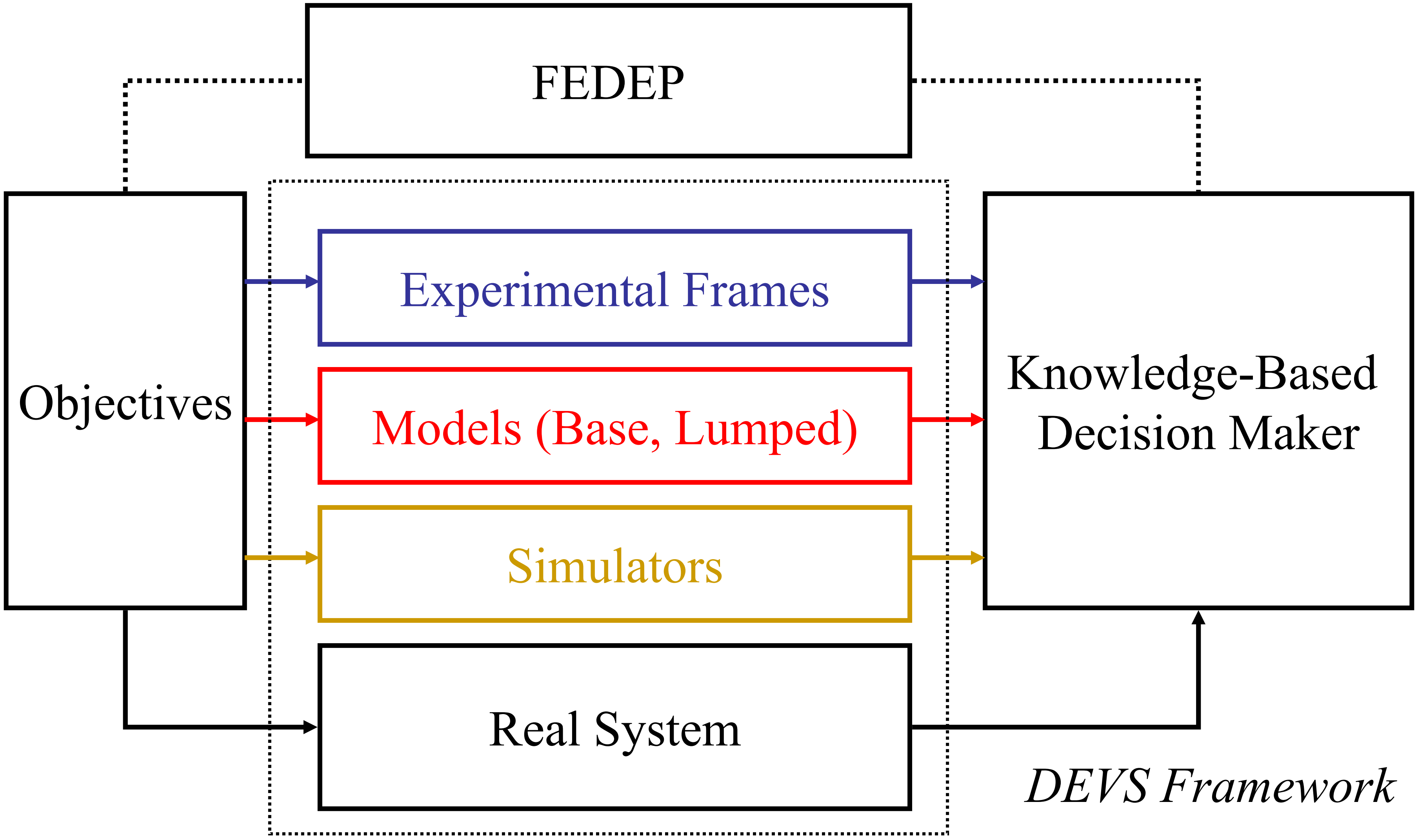
- FEDEP provides a generic process – it does not provide details of how to execute it
  - no underlying structure is prescribed to tie the steps that are required for V&V
- It offers a conceptual evolutionary sequence without specifying mechanisms for transitioning from one step to another
- It does not include a semantically rich M&S framework that can provide relationships to link the various process steps elements together

VV&A and New M&S Development

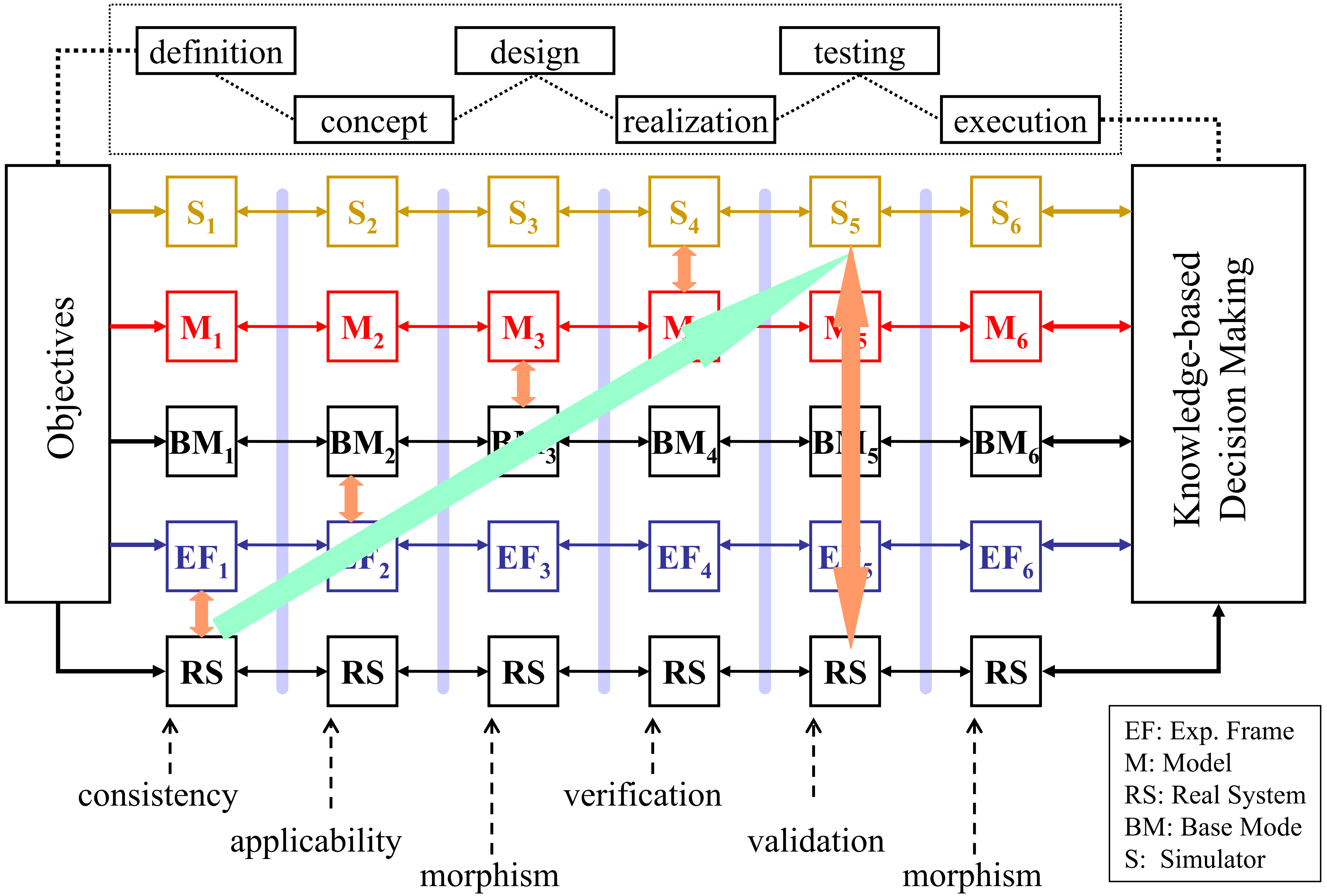


source: http://www.msiac.dmso.mil/vva/

Approach to Merging the FEDEP with the M&S Framework



FEDEP Approach to V&V: Refined by M&S Framework



Layered Architecture for M&S

Cooperation among participants	Collaboration
Decision Making in Application domains	Decision
Searches, Evaluates Design Space	Search
Specifies Dynamics	Modeling
Interprets Models	Simulation
Allocates Resources & Mediates Processes	Middleware
Supports Software and Hardware aspects of M&S	Network

EF and Model Specifications

FEDEP Step	EF Specifications	Model Specifications
1	EF_1: Requirements level	M_1: Observation system
2	EF_2: Conceptual level	M_2: IORO & IOFO systems
3	EF_3: Design/realization level	M_3: IO system
4	EF_4: Design/realization level	M_4: Coupled system
5	EF_5: Parameterized experimentation level	M_5: Parameterized coupled system
6	EF_6: Parameterized experimentation level	M_6: Parameterized coupled system

derivability

applicability

Separating EF and Model Specifications

Experimental Frame

- Requirements Level
  - $\langle \mathbf{T, I, C, O} \rangle$
  - T: time base
  - I: a set of variables, the input variables
  - C: a set of variables, the run control variables
  - O: a set of variables, the output variables

Model

- Observation System
  - $\langle \mathbf{T, I, O} \rangle$
  - T: time base
  - I: a set of variables, the input variables
  - O: a set of variables, the output variables

applicability relation

⇒ separation of concerns is essential to manage explosion of model behavior due to wide range of conditions and choices

⇒ time, input, and output variables must be specified w.r.t. one another; the resolution (degree of accuracy) of these variable, however, may not necessarily be identical.

Experimental Frame Specification

- Requirements Level
  - $\langle \mathbf{T}, \mathbf{I}, \mathbf{C}, \mathbf{O} \rangle$
- Conceptual Level
  - $\langle \mathbf{T}, \mathbf{I}, \mathbf{C}, \mathbf{O}, \Omega_{\mathbf{I}}, \Omega_{\mathbf{C}}, \mathbf{SU} \rangle$
    - Ω_I : admissible input segments
    - Ω_C : admissible control segments
    - SU : a set of summary mappings

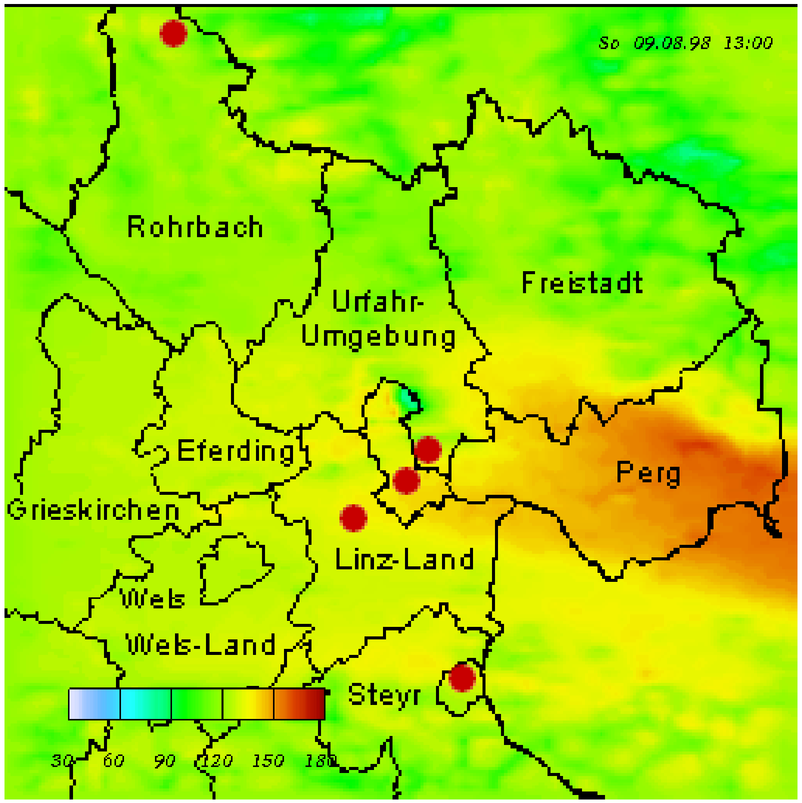
EF-Specification (cont.)

- Design Level
  - $\langle \mathbf{T}, \mathbf{I}, \mathbf{C}, \mathbf{O}, \Omega_{\mathbf{I, N}}, \Omega_{\mathbf{C, N}}, \mathbf{SU_N} \rangle$
    - $\Omega_{\mathbf{I, N}}$: Crossproduct of input segments : Generator(s)
    - $\Omega_{\mathbf{C, N}}$: Crossproduct of control segments : Acceptor(s)
    - $\mathbf{SU_N}$: Resultant behavior space : Transducer(s)

- Parametrized System Levels
  - $\langle \mathbf{T^p}, \mathbf{I^p}, \mathbf{C^p}, \mathbf{O^p}, \Omega^{\mathbf{p}}_{\mathbf{I, N}}, \Omega^{\mathbf{p}}_{\mathbf{C, N}}, \mathbf{SU^p_N} \rangle$

Application Example – Following FEDEP++ to create an M&S system for real-time prediction of smog levels in urban regions

- Mesoscale Ozone Forecast Model
- Simulation of a realistic summer smog episode in Greater Linz area/Austria



R. Freigassner, et al., (Systems Analysis and Simulation, GMD-FIRST, Berlin)

EF-Ozone Forecast Model

- Question
  - Within a given time period, do a minimum number of measuring stations exceed an acceptable smog health risk threshold level?
- Step 1
  - Choose input variables: daily wind, polution (vehcile emissions), and cloud cover
  - Choose output variables: number of warnings and hazard level
  - Choose control signals

EF- Ozone Forecast Model (cont.)

- Step 2
  - Determine I/O data space (trajectories)
  - Determine control regimes (trajectories)
  - Determine mappings
- Steps 3-4
  - Develop generators for wind, cloud cover, daily variations
  - Develop acceptors for initialization and termination conditions for polution
  - Develop transducers for observing a set of measurement stations for smog, vehicle polution, ...
- Steps 5-6
  - Determine bounds based on parameterization

Assessment of FEDEP++

- FEDEP++ provides a process with some details of how to execute it
- It is based on a sound framework for M&S that provides relationships to link the elements together
- M&S Framework provides an underlying structure to tie different steps that are needed for V&V
- Suggests a process as well as places in which mechanisms for iteration would be helpful
- Suggests why V&V is inherently a complex task and the kinds of tools that would help to improve its execution

Activity – Generic Characteristic of Complex Systems

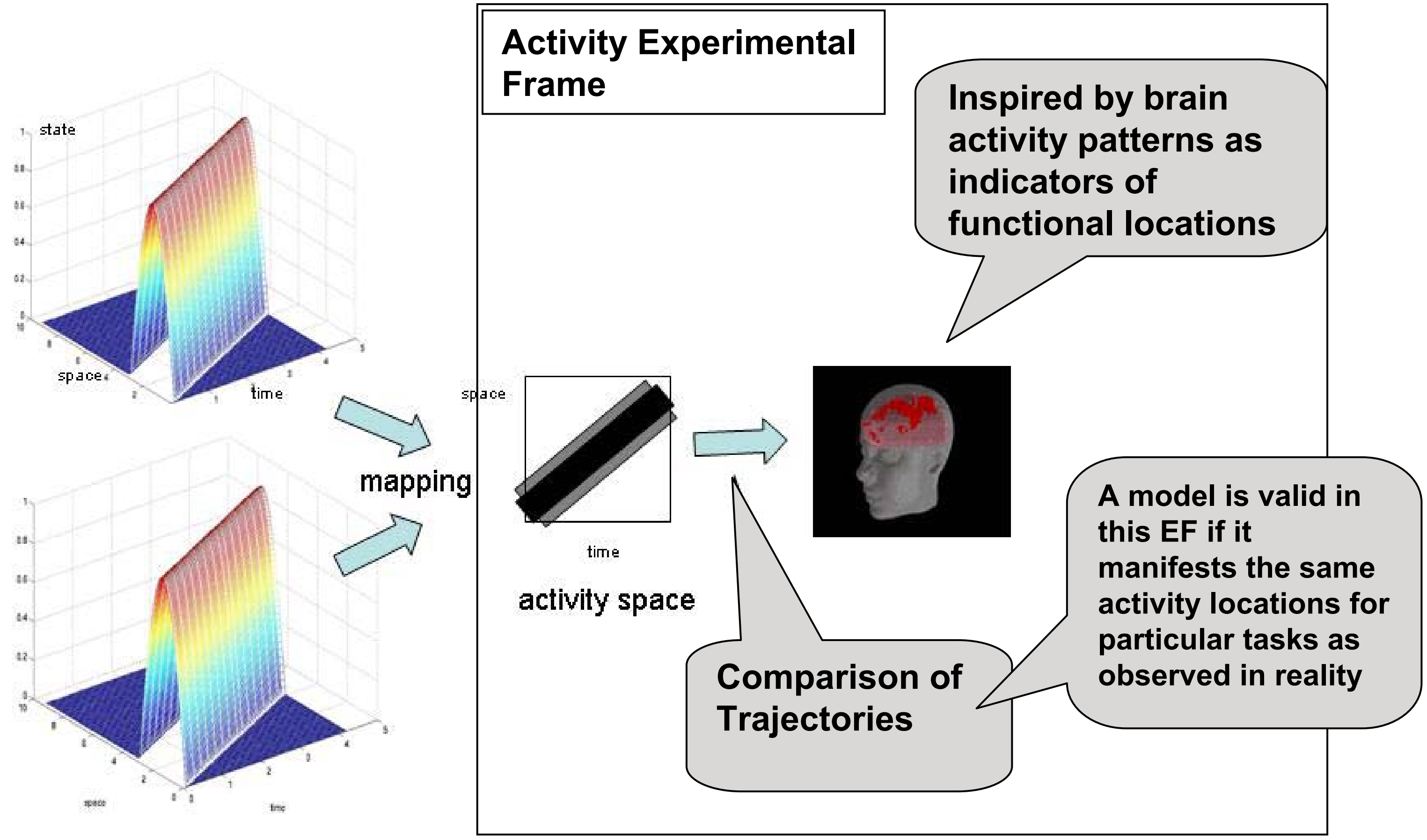
- Activity definition and measurement
- Correlated activity – avalanches in Self Organized Criticality
- Relation to computational efficiency
- Relation to V&V

Requirements for an Activity Measure

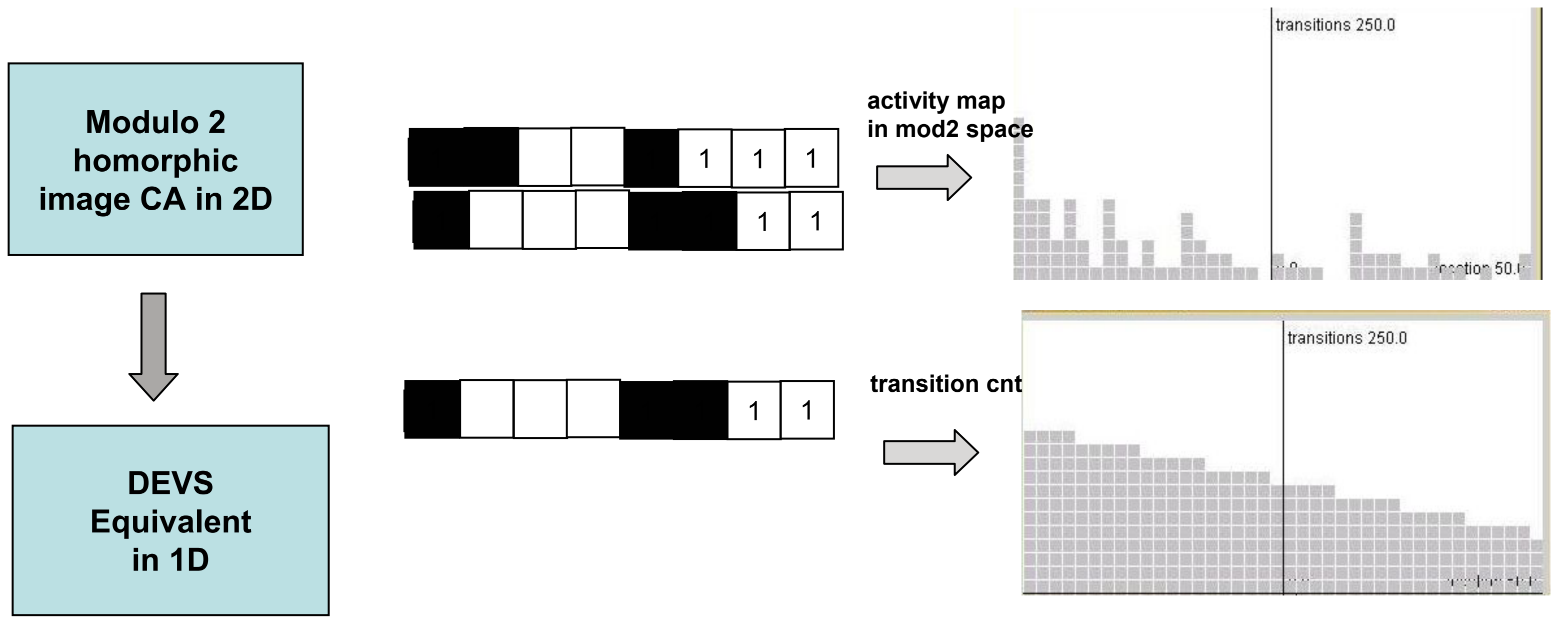
Need a measure of activity that is:

- generic in systems theory spirit – allowing application to a variety of model domains
- supports mapping into activity space enabling comparison of complex system trajectories
- computationally feasible - allowing its use to compare trajectories online, i.e., while simulation is in progress
- computationally relevant- allowing its use to direct computational resources toward regions in space and time that have high activity

Mapping large scale system trajectories into Activity Space


state
space
time
mapping
Activity Experimental Frame
space
time
activity space
Inspired by brain activity patterns as indicators of functional locations
Comparison of Trajectories
A model is valid in this EF if it manifests the same activity locations for particular tasks as observed in reality
space
time

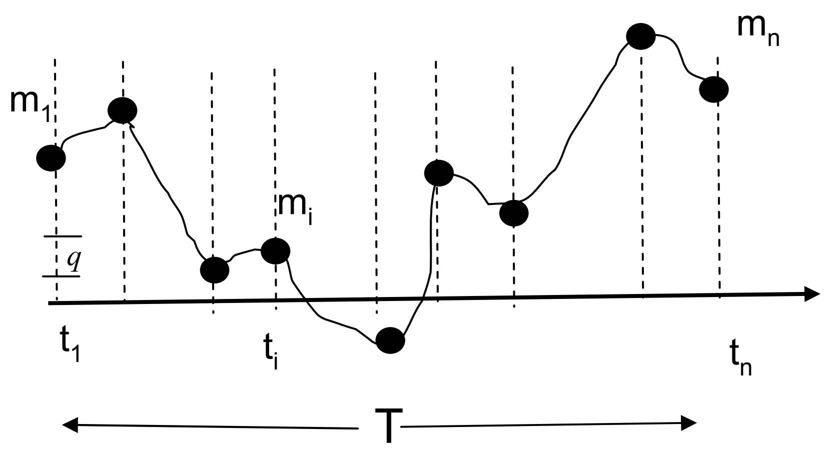
Example: Activity Relationships in Pascal Triangle



- •number of actual transitions of DEVS model vs number of true state changes
- •Activity pattern -- both decrease with cell index. The extra event checking required by the DEVS accounts for larger area.

Definition and Measurement of Activity

Continuous segment



$$ActivityPattern(T) = (m_1, t_1)\ldots(m_i, t_i)\ldots(m_n, t_n)$$

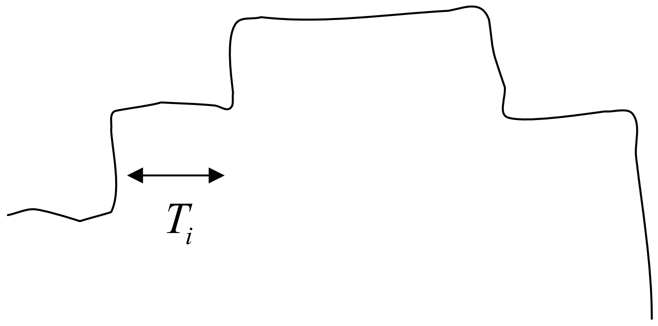
$$Activity(T) = \sum |m_{i+1} - m_i| \quad AvgActivity(T) = Activity / T$$

$$NumberOfThresholdCross(T, q) = Activity(T)/q$$

$$AverageDerivative(t_i, t_{i+1}) = |m_{i+1} - m_i| / t_{i+1} - t_i$$

activity in monotonic region reflects avg rate of change

Piecewise Continuous Segment

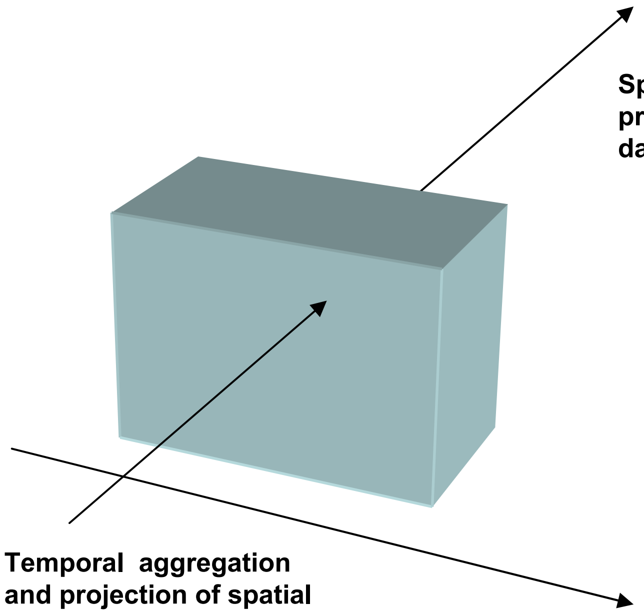


$$Activity(T) = numberOfDiscontinuities(T) + \sum_i Activity(T_i)$$

Activity = sum of ranges.
Avg Activity ~ A*f
for wave with amplitude A
and frequency f

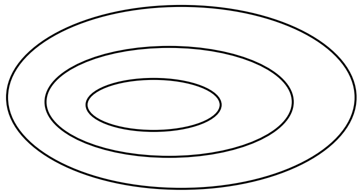
computational efficiency occurs when number of transitions reflects number of threshold crossings

Activity Patterns in Complex Systems

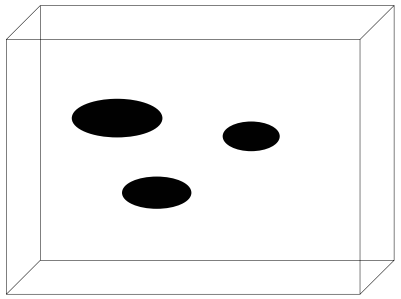


Spatial aggregation and projection of raw activity data

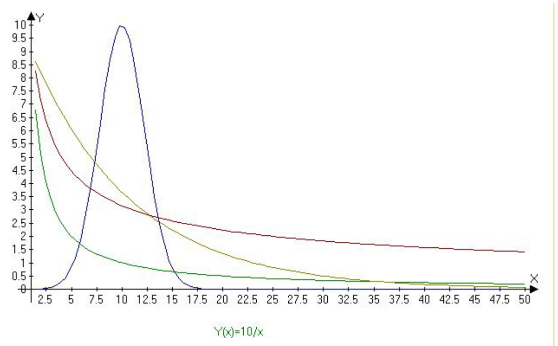
Temporal aggregation and projection of spatial data



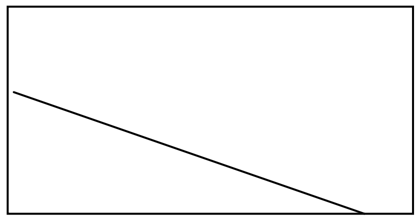
shock wave



localized centers

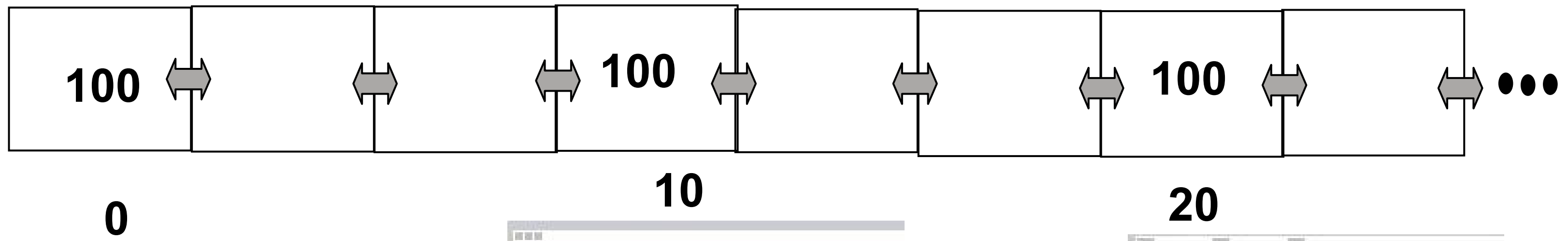


temporal distribution



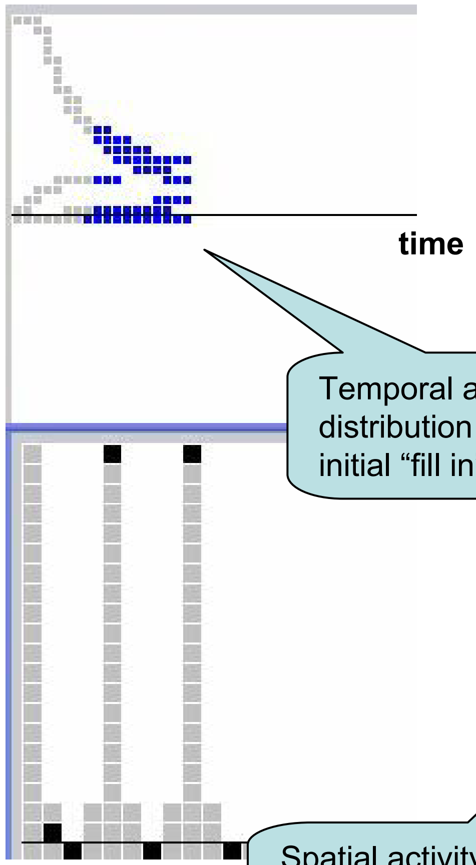
spatial steady state distribution (e.g. Pascal Triangle)

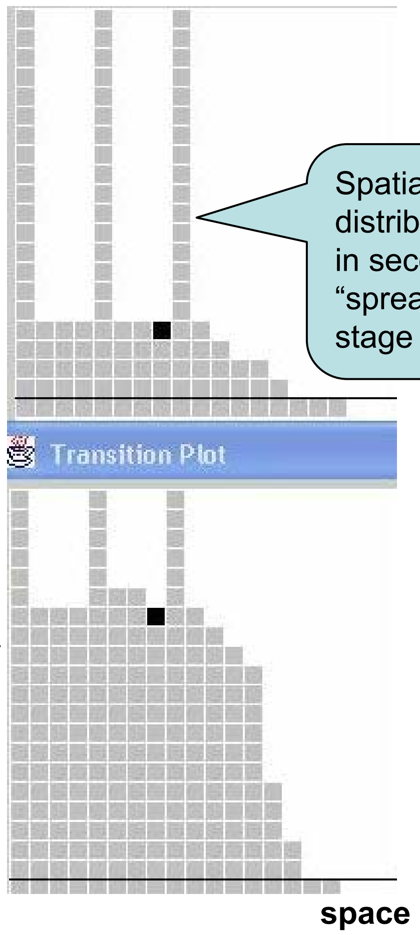
Example: Activity Patterns for One Dimensional Diffusion (PDE)



•One-dimensional diffusion – region between spikes first fills out. Later this region diffuses out to the right.

•This PDE is simulated using DEVS – the number of transitions reflects the measured activity, indicating efficient simulation





Cellular DEVS Models of Spatial Threshold Systems

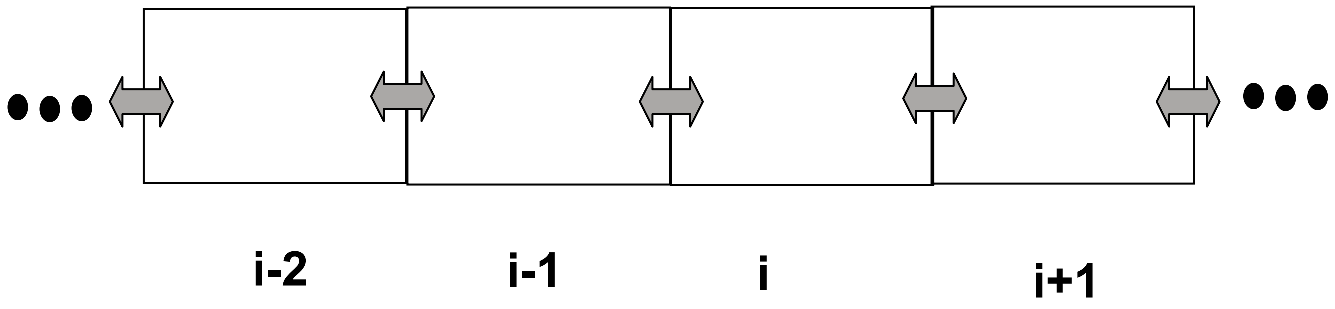
DEVS abstractions of continuous systems with threshold behavior allow simulations that would otherwise be too complex/time consuming to be feasible

An interesting class of such systems are the Self Organized Criticality (SOC) natural systems which depend on threshold properties of continuous systems

These properties allow using DEVS to model them with appropriate abstractions

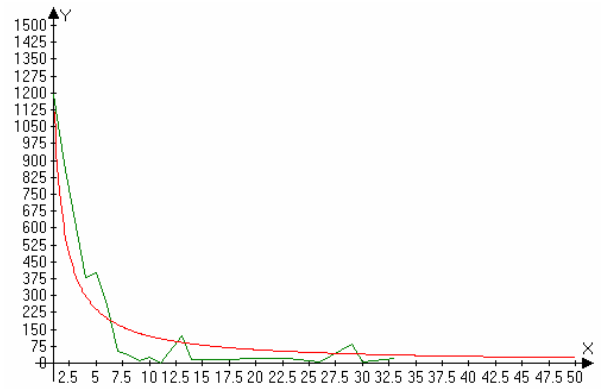
We break space into cells and model these with DEVS models; the cells are placed into 1,2,3 or higher dimensional grids and coupled with neighborhoods form a cellular space.

SOC Earthquake expressed as a One Dimensional Cellular Space



Due to compression on tectonic plates, each cell accumulates energy at a constant rate until a critical threshold is exceeded. At this point a fraction of its accumulated energy is equally distributed among its neighbors. This reduces the time to achieve their critical levels. If the additional energy causes a cell to exceed

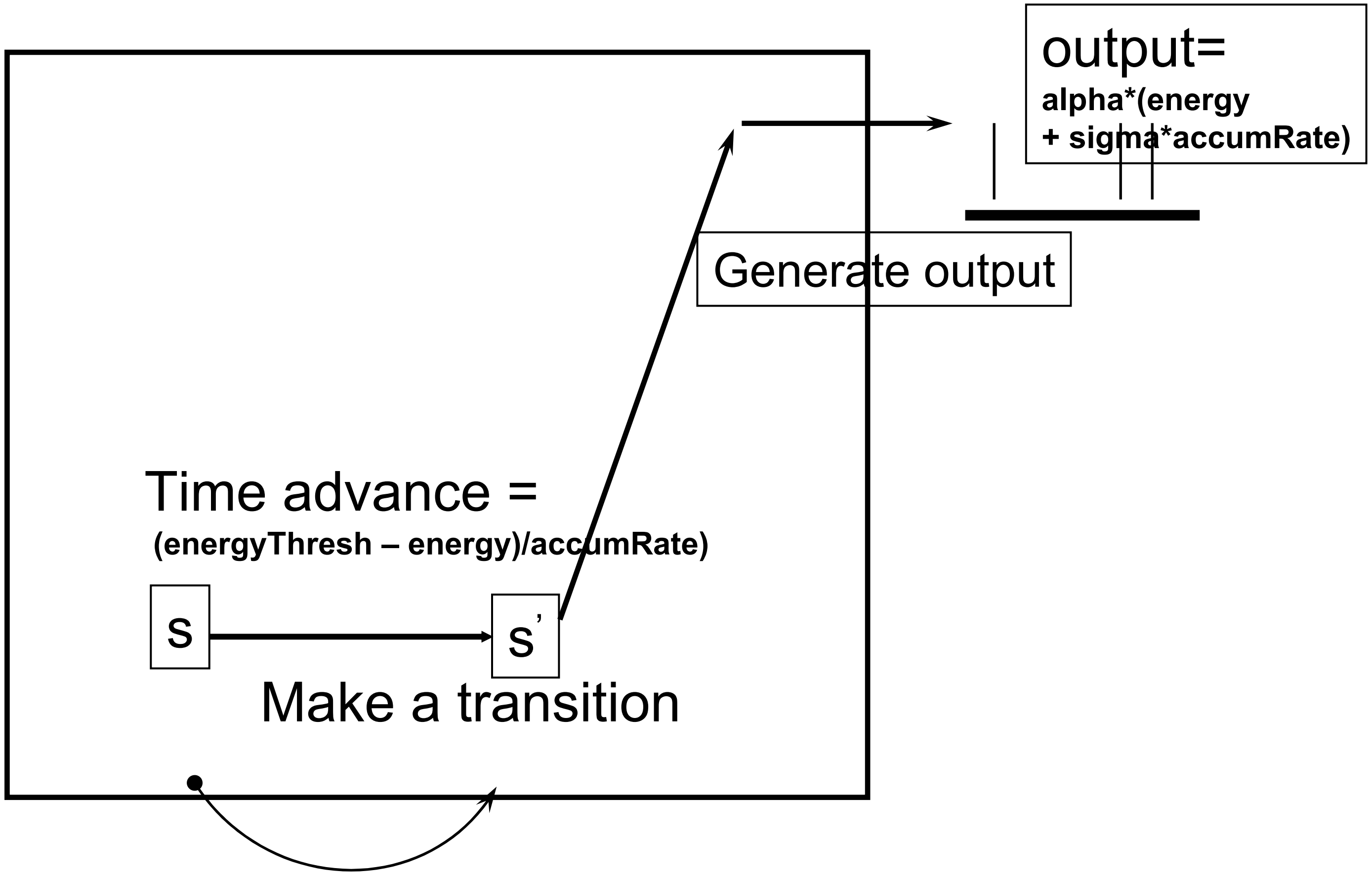
its threshold, then it will release energy to its neighbors. The size of an avalanche is the number of such propagated releases that occur until all cells are below the threshold. Releases occur with ta = 0, so the size of an avalanche is the length of a transitory sequences of external transitions.



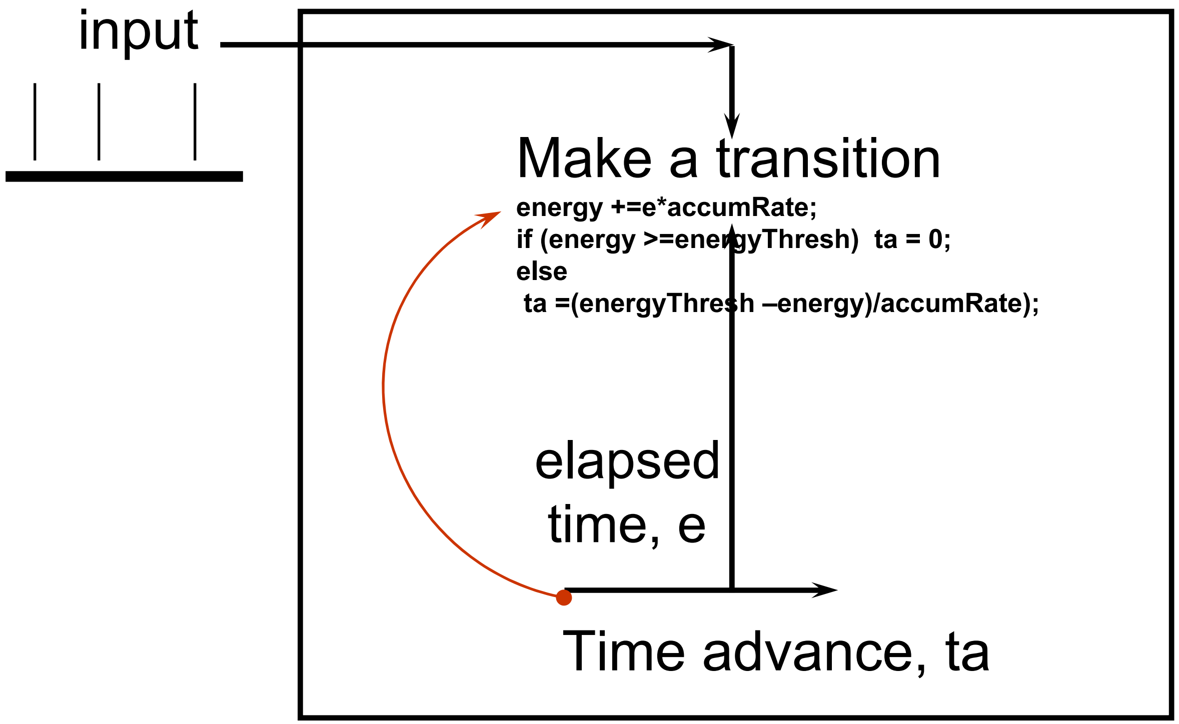
Power law distribution of avalanches (~1/x)

Per Bak, "How Nature Works"
H. Jensen, "Self Organizing Complexity"

Internal Transition /Output Generation -- a cell reaches its threshold energy and distributes its energy to its neighbors; the cell starts accumulating energy starting from zero



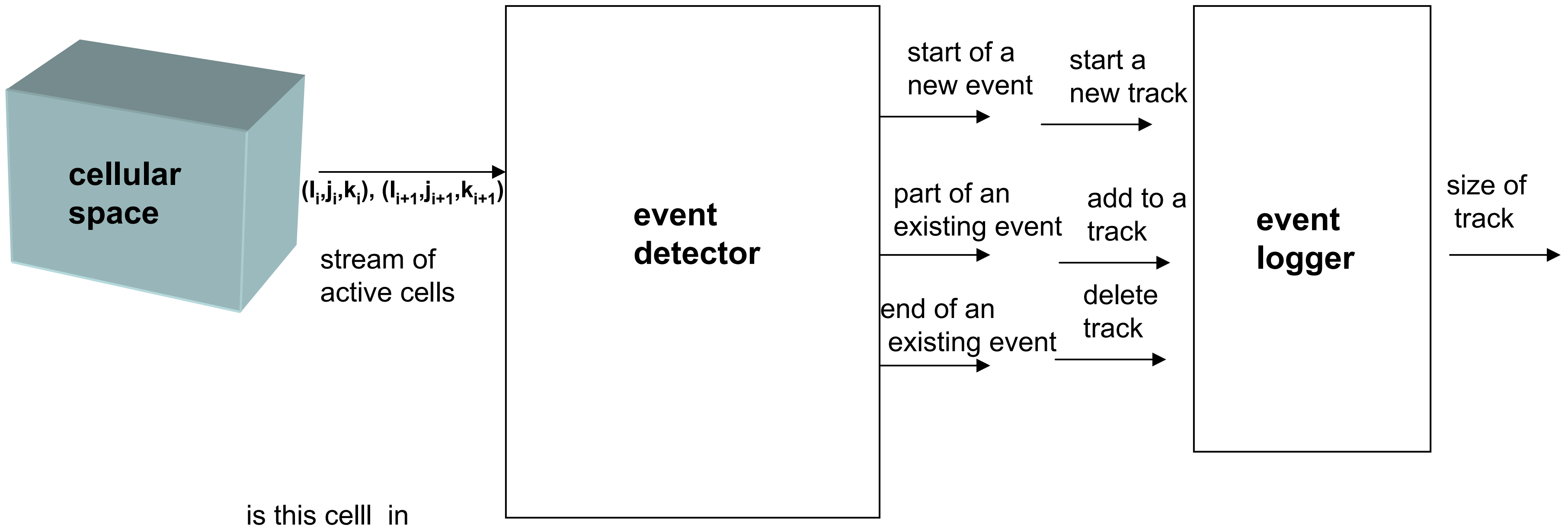
Response to External Input – a neighbor cell receives energy and schedules itself to reach the threshold earlier than before (or it may exceed threshold and become critical)

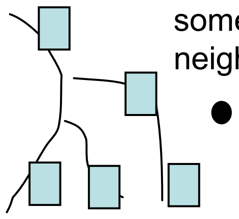


```
energy +=e*accumRate;
if (energy >=energyThresh)  ta = 0;
else
 ta =(energyThresh –energy)/accumRate);
```

Experimental Frames for SOC

Causality detection in cellular space – like tracking of objects by radar





is this celll in some event neighborhood?

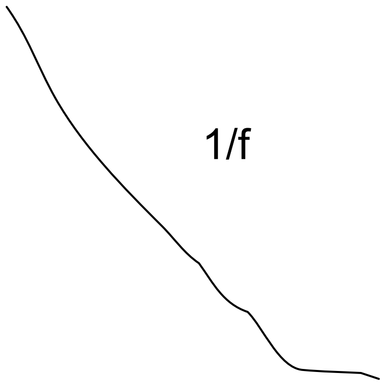
(I,j,k) is part of an existing
event if it is a neighbor of some
cell In that track - ask all tracks: is this
in your neighbor hood?

(I,j,k) starts a new event if it is not
part of any event

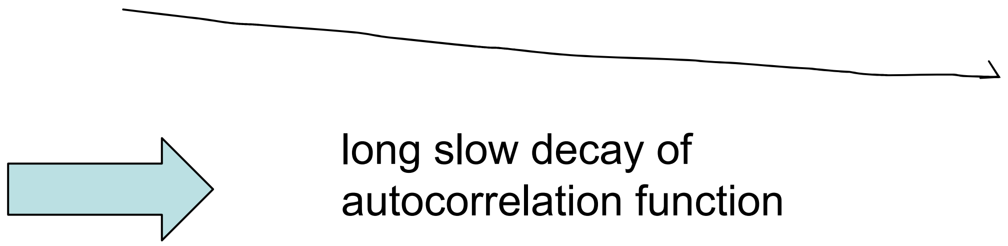
when does an event end? – need further hints

event = avalanche
= radar track

The meaning of 1-over-f noise (Power Spectrum Decay)



one-over-f decay of
power spectral density function



long slow decay of
autocorrelation function

$$\omega = 2\pi f$$

$$PSD(\omega) = \int G(t) e^{i\omega t} dt$$

$$Let \quad G(t) = e^{-at}$$

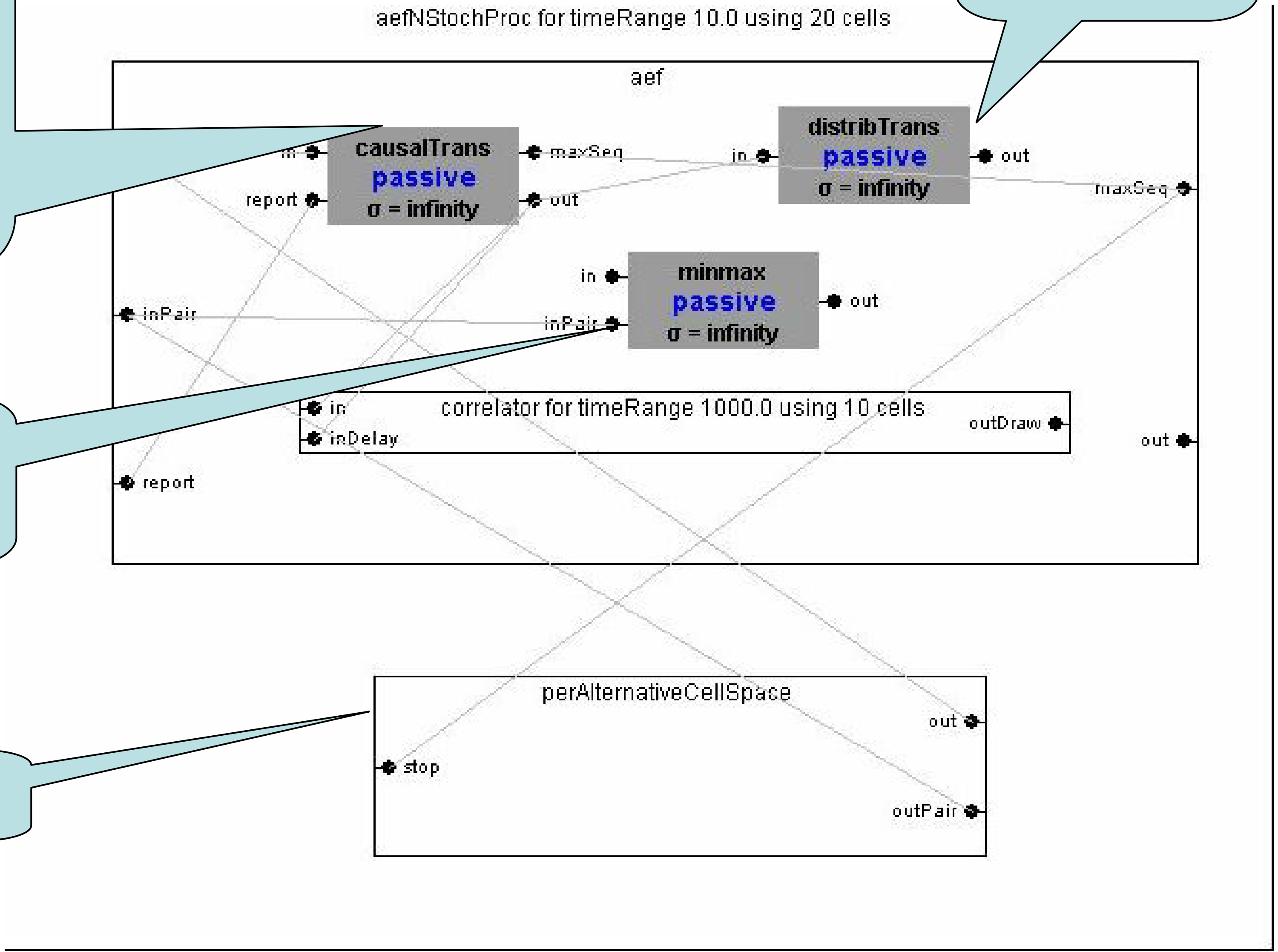
$$PSD(\omega) = \sqrt{\frac{1}{\omega^2 - a^2}} \cong \frac{1}{\omega} \, for \quad a \to 0$$

$$G(t) = \int F(s)F(s+t)ds - (\overline{F})^2$$

$$G(0) = \overline{F^2} - (\overline{F})^2 = \mathrm{var}(F)$$

$$G(\infty) = (\overline{F})^2$$

Experimental Frame for Avalanche Observation



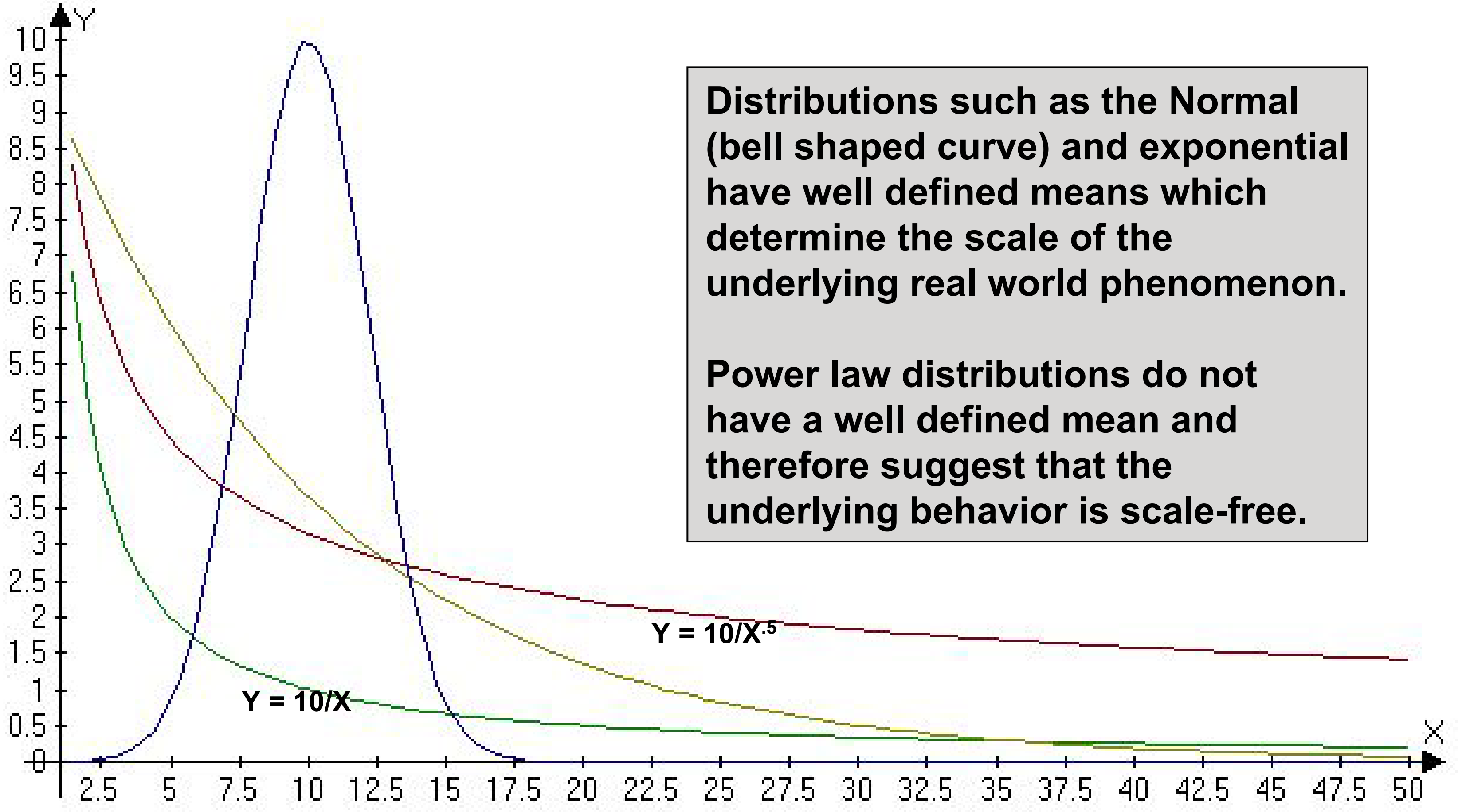
Causality Transducer – observes cell id input stream and segments it into causal chain events (avalanches). Causality is determined by neighborhood adjacency.

Distribution transducer keeps running tally of event (avalanche) sizes

Minmax keeps track of state distribution; computing running min, max, and avg

Cellular model under investigation

The Meaning of Power Law Distributions

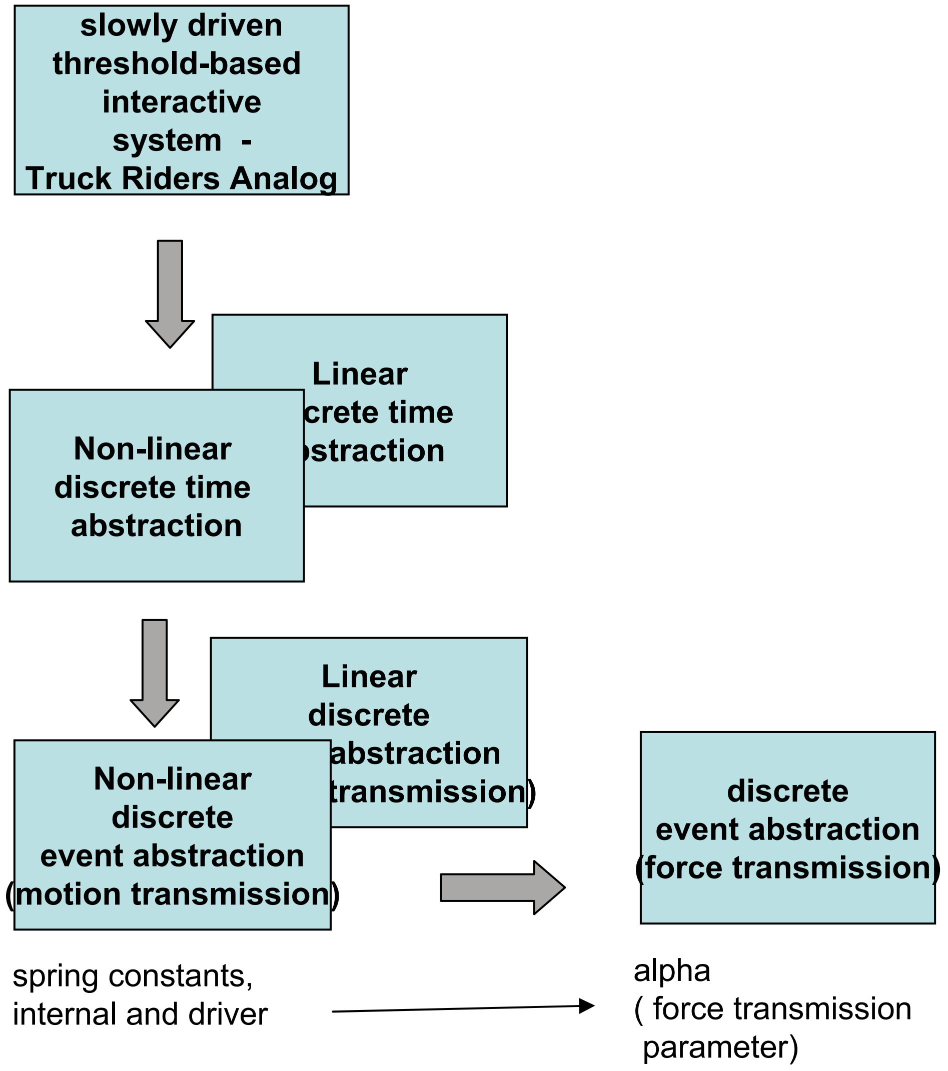


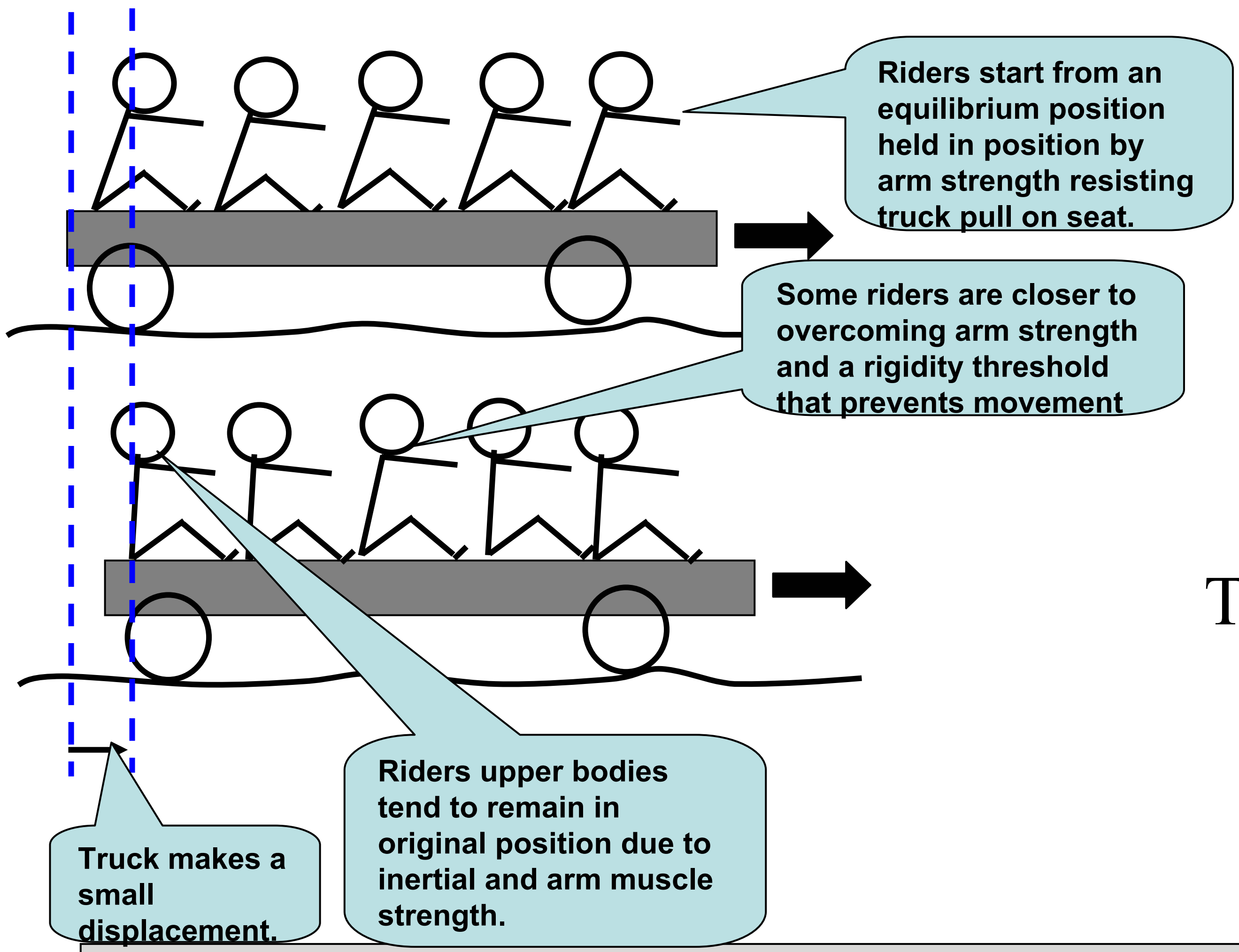
Distributions such as the Normal (bell shaped curve) and exponential have well defined means which determine the scale of the underlying real world phenomenon.

Power law distributions do not have a well defined mean and therefore suggest that the underlying behavior is scale-free.

Y(x)=10/x
Y(x)=10/(x^.5)
Y(x)=10*exp(-x/10)
Y(x)=10*exp(-.1(x-10)^2)

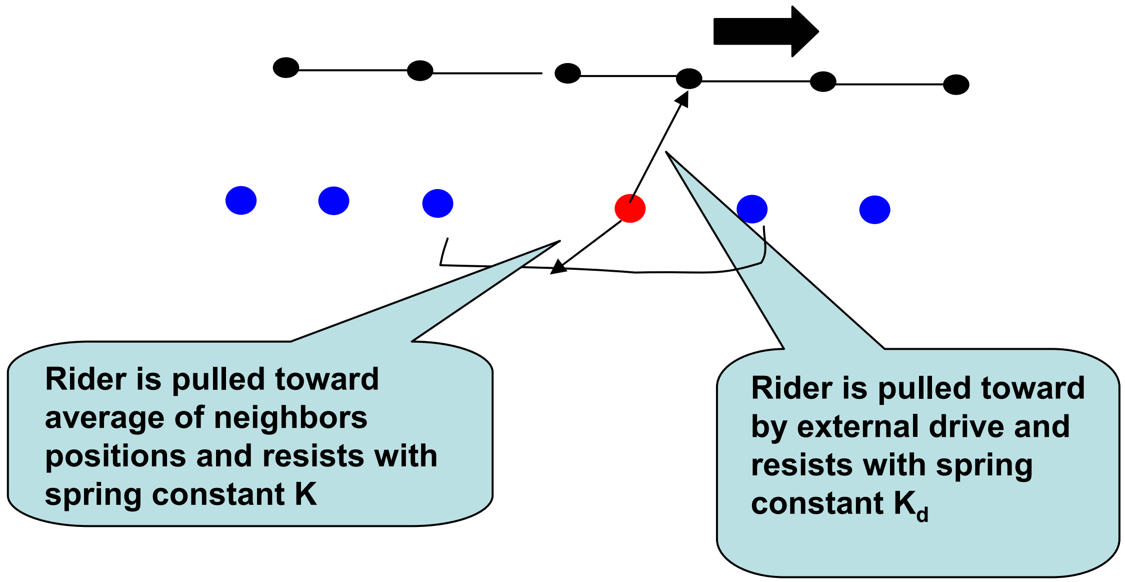
Models for Earthquake SOC


slowly driven threshold-based interactive system - Truck Riders Analog
Linear crete time ostraction
Non-linear discrete time abstraction
Linear discrete abstraction transmission)
Non-linear discrete event abstraction (motion transmission)
discrete event abstraction (force transmission)
spring constants, internal and driver
alpha (force transmission parameter)


Riders start from an equilibrium position held in position by arm strength resisting truck pull on seat.
Some riders are closer to overcoming arm strength and a rigidity threshold that prevents movement
Riders upper bodies tend to remain in original position due to inertial and arm muscle strength.
Truck makes a small displacement.

Truck riders analog

- Riders sitting hand upon shoulder on a moving truck try to resist being displaced forward or backward. However, the forward motion of the truck must eventually require that each rider achieves its speed.
- Small bumps of the road cause riders to be randomly distributed in closeness to truck speed. At any time some riders are closer to achieving the truck speed than others.
- When one such rider is unstuck and lurches forward to match truck speed (at which point s/he experiences no net force), s/he pushes on the rider in front and pulls the rider in back, bringing them closer to truck speed.
- An avalanche occurs if several neighbors are close to criticality (truck speed) and relaxations to truck speed started by one propagate instantaneously to others.
- Avalanches are only possible if the truck pull on riders is not too large relative to their arm strength. Otherwise all riders will be quickly brought up to equilibrium position.


Rider is pulled toward average of neighbors positions and resists with spring constant K
Rider is pulled toward by external drive and resists with spring constant K_d

Total force on rider is

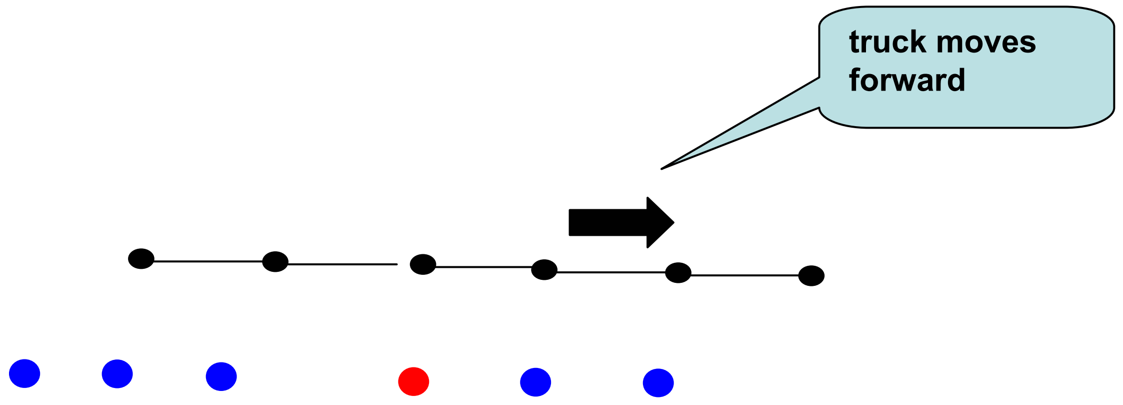
$F(t) = 2K*(avgNeighPos(t) – pos(t)) + K_d *(driverPos(t) – pos(t))$

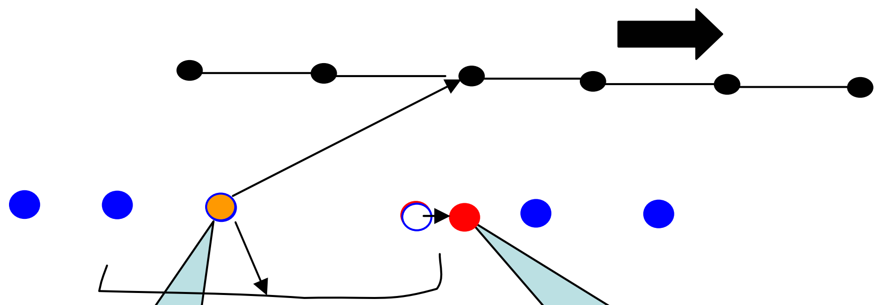
where pos(t) is the position of the rider at time t, and

driverPos(t) = initialDriverPos + v*t (where v is the constant speed).

Note avgNeighPos(t) remains fixed at its last value until a neighbor moves
Total force increases until it exceeds the static friciion threshold

At this time, t’ the rider moves to a new position, pos(t’), that satisfies F(t’) = 0)
i.e., 2K*(avgNeighPos(t) – pos(t’)) + Kd *(driverPos(t) – pos(t’)) = 0
where t was the last update time before t’.
This new rider’s position is sent to the neighbors and alters the total force on each. Meanwhile, each rider is going through the above cycle.


truck moves forward



this increases the total force on each neighbor pulling it toward its equilibrium position. The increase in force is alpha times the original total force on the just moved rider, where alpha = $K/(2K + K_d)$

when total force on rider exceeds static friction (threshold), it moves to position that nullifies total force on it

The time scale is Thresh/(speed*Kd) which sets the energy threshold

The derrivation

$$\text{alpha} = K/(2K + K_d)$$

provides a parameter mapping from the truck rider model to the earthquake model. We verify that the external drive spring constant cannot be too large (i.e., alpha too small) in relation to the internal spring constant K for avalanches (earthquakes) to be possible. On the other hand, the largest value of alpha is ½ for which $K_d = 0$. Here avalanches are also not possible since the moving drive cannot exert a pulling force on the riders. Thus, this model suggests limits on the parameter alpha of the earthquake model.

Basic Relationships:

Each cell (rider) experiences a force which is the total of the spring orces exerted by its neighbors and the drive (moving truck):

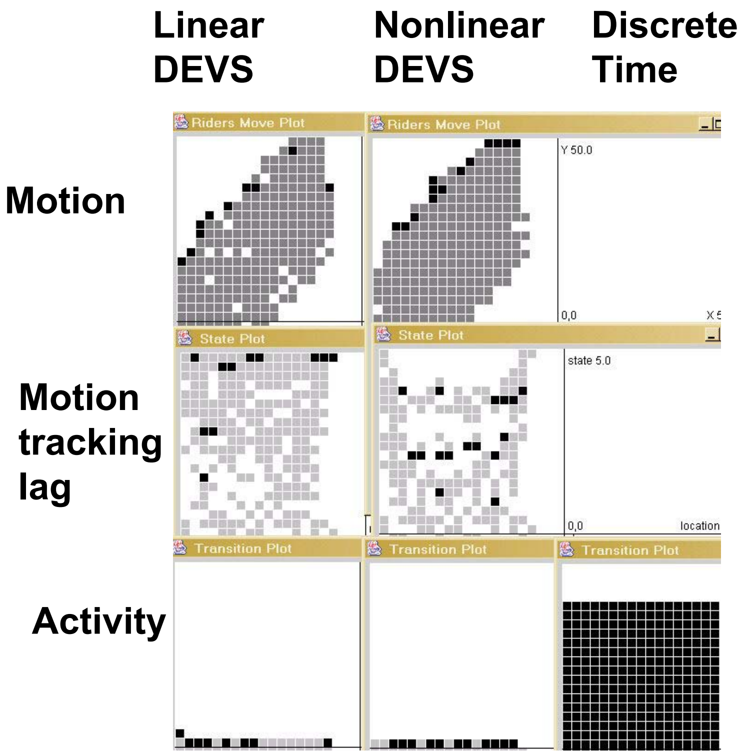
Total Force (myPos, rightPos, leftPos, drivePos) –

Equilibrium position – eqPos satisfies

Total Force (ePos, rightPos, leftPos, drivePos) =0

eqPos (rightPos, leftPos,drivePos) = this cell's position that would nullify the force on it a function of current values of neighbors and drive positions.

Solve by following gradient down from initial estimate given by linear solution.



The discrete time model --

```
until (termination condition):
advance each cell's time by delta
drivePos = drivePos + speed*delta
compute the increased force = TotalForce(…)
if the force exceeds threshold
  then  set myPos = eqPos(rightPos,
leftPos,drivePos)
     (jump immediately to eqPos)
   send Pos to neighbors
```

The discrete event model computes time advance to reach Threshold, ta satisfies

Total Force (myPos, rightPos, leftPos,drivePos+ta*speed) =Threshold

Solve by doing "internal simulation" –

```
timeToReachThreshold(…){
e = 0;
while (Total Force(…,drivePos +e*speed) < Threshold)
   e = e+delta;
 return  e;}
```

Model-Experimental Frame Relationships for Earthquake SOC Domain

$\Rightarrow$ **derivability**

$\Downarrow$ **morphism**

Output / Model	Continuous Motion on multiple time scales	All-at-once Motion to local equilibrium	SOC – Avalanche Power Law Distributions	Speed/Accuracy
Slowly Driven Threshold-based Interactive Model	√	√	√	Unfeasible
Truck Riders Analog Discrete Time		√	√	Painfully slow – due to small time step for threshold crossing detection
Truck Riders DEVS Abstraction (Motion Transmission)		√	√	Fast – a small subset is involved in threshold crossing detection
Original Earthquake Model (Force Transmission)			√	Fast – Force Transmission introduces instability for small cells numbers

Summary and Conclusions

- **System Theory has been combined with object-orientation to provide a sound M&S framework for incremental, evolutionary VV&A**
  - Multifaceted formalization of experiments and models in a unified framework is necessary to achieve specification and development of M&S tools with VV&A capabilities
  - DEVS and HLA offer complementary capabilities toward developing VV&A aware M&S environments

- **FEDEP++ suggests using system-theoretic concepts, constructs, and methods to enable characterization and implementation of VV&A across the FEDEP process**

  - Separation of concerns in terms of models, experimental frames, and simulators w.r.t. real-systems is necessary to achieve compartmentalizing levels of details and transitioning among such levels given the need to develop alternative, often complementary, simulation models

Summary and Conclusions (cont'd)

- **Can begin handling the scale and complexity traits inherent in complex M&S using multi-level specifications for experiments, models in concert with simulators and real-system data**
  - Homomorphism and other equivalence relationships provide rigorous capabilities for quantifiable VV&A measures
  - formalizing interoperability, composability, and their interdependence serves as enablers for VV&A

- **Development of generic activity space concepts within general systems theory allows application to a variety of model domains**
  - computationally feasible – allows compare trajectories online, i.e., while simulation is in progress
  - computationally relevant- allows directing computational resources toward regions in space and time that have high activity

For more literature, software, tutorials,…

Arizona Center for Integrative Modeling & Simulation (ACIMS)

www.acims.arizona.edu